INTERSECTION DETECTION AND NAVIGATION FOR AN AUTONOMOUS GREENHOUSE SPRAYER USING MACHINE VISION

By

PAULO YOUNSE

A THESIS PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF ENGINEERING

UNIVERSITY OF FLORIDA

2005

This thesis is dedicated to my family, Dr. Burks, and all the students in the Agricultural and Biological Engineering Department for enriching both my professional and personal lives with knowledge, confidence, motivation, and personality.

ACKNOWLEDGMENTS

I would like to thank Dr. Thomas Burks for providing me with the opportunity to further my studies and experience in the field of robotics. His support and continuous guidance kept me motivated and on task throughout the period of my graduate study.

I would also like to share my gratitude with Dr. Michael Hannan, Mike Zingaro, Greg Pugh, and Vijay Subramanian for their technical support and availability to offer a lending hand when needed. Lastly, this program would not have been possible without the sponsorship of the National Foliage Foundation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Abstract of Thesis Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Master of Engineering

INTERSECTION DETECTION AND NAVIGATION FOR AN AUTONOMOUS GREENHOUSE SPRAYER USING MACHINE VISION

By

Paulo Younse

August 2005

Chair: Thomas F. Burks
Major Department: Agricultural and Biological Engineering

Vehicle automation is a growing interest among the agricultural community. Design of a robotic greenhouse sprayer can provide more accurate spraying of pesticides and fungicides, reduce operational costs, and decrease health risks associated with human exposure to dangerous chemicals. Building a fully autonomous vehicle requires a reliable navigation system to guide the vehicle through a greenhouse, as well as detect and negotiate intersections. Simple and low-cost sensor arrangements, such as a system based on a single CCD camera, are required to generate a feasible economic solution to vehicle navigation in agriculture.

A visual navigation system capable of tracking a path, detecting intersections, and navigating through intersections was developed for a six-wheel differential steering vehicle. The system was designed for a corridor environment and can be applied to tasks such as autonomous greenhouse spraying and robotic citrus harvesting.

Path navigation was accomplished using digital images taken from a CCD camera mounted on a robotic greenhouse sprayer. Intersection detection was accomplished by first classifying pixels of an image as path or non-path. Left and right path edges were determined from the threshold image using least squares fitting. Path pixels extending beyond a specified distance from the left and right edges indicated an approaching intersection. The distance from the vehicle to the intersection was estimated based on the set position and orientation of the camera using projective geometry. Intersection navigation was accomplished by 1) tracking ground features to guide the vehicle to a desired position in the intersection, 2) proceeding with a turn, and 3) concluding when the vehicle was aligned with the next path.

Testing verified the accuracy of the vehicle's camera model, visual odometer, ability to detect intersections, and ability to navigate intersections. Results were obtained for the vehicle making a turn and driving through intersections lined with tape and intersections lined with potted plants.

CHAPTER 1
INTRODUCTION

This thesis addresses the use of machine vision for intersection detection and navigation of a robotic sprayer through a greenhouse. Tasks involved in the process include image segmentation of the path, vehicle control down the path, intersection detection, and intersection navigation. Additional items developed include a camera model and a visual odometer. The techniques investigated in this thesis can also be applied to robot navigation through orchards, citrus groves, and corridor environments.

Background and Motivation

Vehicle automation is a growing interest among the agricultural community. Many feasibility studies were made on autonomous agricultural vehicles. Have et al. (2002) investigated the development of autonomous weeders for Christmas tree plantations. Design requirements for such a vehicle were explored, specific behaviors for navigation and operation defined, and a system architecture proposed. Hellström (2002) performed a similar study for autonomous forest machines as part of a project to develop unmanned vehicles that transport timber.

Robotic citrus harvesting is gathering support in the citrus community, due to an expected difficulty in acquiring seasonal labor and increased international competition in the citrus market (Hardy, 2003; Leavy, 2002). Several robotic tree-fruit harvesters were developed, but their crop recovery and picking rates were too low to justify their cost (Hardy, 2003). Another application of autonomous agricultural vehicles is autonomous greenhouse spraying. Benefits of an autonomous greenhouse sprayer include increased

accuracy and precision with spraying, which would contribute to more efficient use of resources, capability to operate 24-hours a day, and decreased health risks associated with human exposure to dangerous chemicals.

Background on the Autonomous Greenhouse Sprayer

Singh (2004) designed a 32 x 16-inch autonomous greenhouse sprayer through the Agricultural and Biological Engineering Department at the University of Florida under sponsorship of the National Foliage Foundation. The vehicle was designed to navigate through 18, 20, and 24-inch aisles. Two 0.75 horsepower DC motors with 20:1 gear reducers powered two separate three-wheel drive trains. Turning was accomplished with differential steering. Vehicle control down the center of test paths was carried out independently using ultrasonic range sensors, ladar, and machine vision with a fuzzy PD controller. The vehicle mounted with the tested sensors is shown in Figure 1-1.

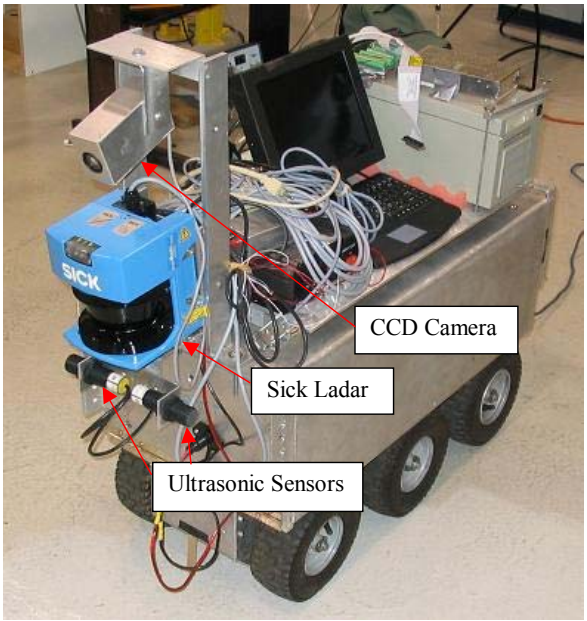


Figure 1-1. Autonomous greenhouse sprayer with ultrasonic, ladar, and CCD camera attached.

Greenhouse Configurations

Most greenhouses grow plants on a series of benches, accessed by aisle ways. Stevens et al. (1994) mentioned that most potted greenhouse crops were grown on benches 32 to 36-inches tall. Aisles between benches were 18-inches wide, and center aisles are 3 to 12-feet wide for equipment. Benches were 3-feet wide if against a wall, or up to 6-feet wide if they can be accessed from both sides.

Schnelle and Dole (1991) stated that benches normally did not exceed 36-inches high. They also mentioned that benches were 3-feet wide if against a wall and up to 6-feet wide if they can be accessed from both sides.

Bartok (1992) recommended 18 to 24-inch aisle widths and benches 3-feet wide if worked from one side, 6-feet wide if worked from both sides.

Kessler (1998) recommended walkways 2 to 3-feet wide, with main aisles 4 to 6-feet wide. For wheelchair access, the aisles could be 4-feet wide. Benches were 2 to 3-feet wide if accessed from one side, 4 to 5-feet wide if accessed from both sides. Benches were normally 24 to 36-inches high. For wheelchairs access, the benches should be 30 to 36-inches high.

Singh (2004) designed an autonomous greenhouse sprayer for 18, 20, and 24-inch aisles. Center aisles were mentioned to be typically 8 to 12 feet wide. Aisles were level, with sand aisles having grades less than 1%. The benches were 24 to 30-inches above the ground. The aisle ground between benches could be sand, concrete, or gravel.

Thesis Objectives

To meet the needs associated with the greenhouse configurations described above, design objectives for vehicle guidance and intersection navigation for an autonomous greenhouse sprayer are:

1. Path Following: The vehicle must be capable of driving through the center of 18 to 24-inch aisles and along single edges for the larger center aisles.

2. Intersection Detection: The vehicle must be able to detect intersections between aisles with the consideration that the benches on these aisles can be as low as 2-feet tall and as high as 3-feet tall.

3. Intersection Navigation: The vehicle must be able to both complete turns and drive through detected intersections.

CHAPTER 2
LITERATURE REVIEW

A navigation system for an autonomous greenhouse robot is required to take on the following tasks:

1. Following a path along greenhouse benches
2. Detecting intersections between aisles
3. Traversing intersections to reach the next desired aisle

Much research documenting mobile robot navigation in both indoor and outdoor environments exists as discussed in DeSouza and Kak (2002). In order to accomplish the tasks of path tracking, detecting intersections, and navigating through intersections, sensors are needed to gather information about the environment. A sense of depth is also necessary to follow edges of the path at specified distances and detect intersections and determine where they are in relation to the vehicle. A path model may also be used to assist with path identification. Due to the vast amount of research accomplished in these fields, only select works will be cited, with a stronger focus on agricultural vehicles.

Path Following

Several primary path following algorithms were used for following a path. Among these were Follow the Carrot (Figure 2-1), Pure Pursuit (Figure 2-2) (Coulter, 1992), and Vector Pursuit (Wit, 2000). Another method called Follow the Past was developed and compared with the Follow the Carrot and Pure Pursuit algorithms in Hellström and Ringdahl (2002).

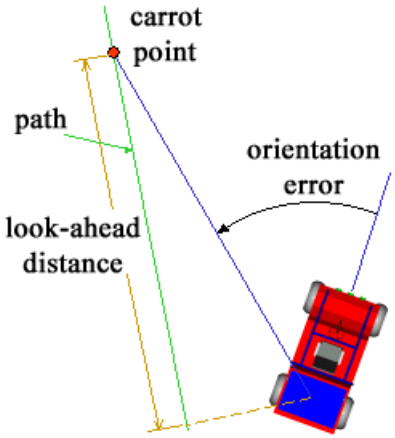


Figure 2-1. Follow the Carrot (University of Florida, 2004).

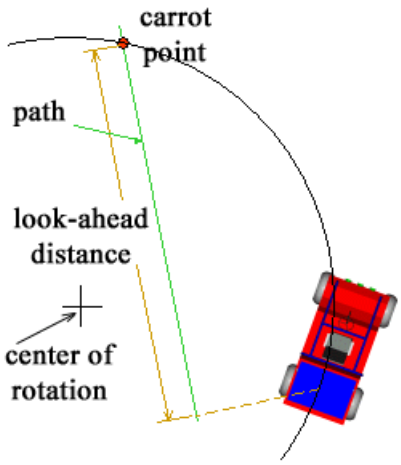


Figure 2-2. Pure Pursuit (University of Florida, 2004).

Other techniques included reducing error of the path center from the vehicle centerline at a specified look-ahead distance, equating the optical flow of both sides of the path, lining the vehicle up with the path vanishing point, and guiding the vehicle towards the largest "free space."

Various control systems were tested for navigation based on the path location and orientation relative to the vehicle. Benson et al. (2003) controlled a harvester with a PID controller. Fehr and Gerrish (1995) used pure pursuit with a proportional gain on the steering angle to control a row crop harvester down a path. ALVINN was a neural

network system that based its steering down a path from actual human responses and driving behavior to training images of the path (Pomerleau, 1995). Reid (1998) reviewed various tractor models that were used for steering controller design.

Below are more descriptive explanations on how researchers followed paths both in corridor, outdoor, and agricultural environments:

Fused Vision-Based Control Signals using Vanishing Points and Optical Flow

The robot described in Carelli et al. (2002) navigated down the center of a corridor by following two control objectives:

1. Aligning its horizontal axis with the hallway vanishing point. The vanishing point was calculated from the intersection of the Hough Transform-derived perspective lines from the edges where the walls meet the floor.
2. Equating optical flow from both walls.

The two control signals were fused using a Kalman filter.

Free Space from Stereovision

Matsumoto et al. (2000) used stereovision to measure the distance from the vehicle to the walls along the path and interpreted this information as free space around the vehicle (Figure 2-3). The vehicle was guided though a hallway by moving towards the direction of the furthest point down the hall.

Horizontal Distance of Crop Row Edge from Camera Center

Fehr and Gerrish (1995) assumed the crop row edge to be perfectly aligned with the center of the camera. The guidance system is initialized by entering the camera's height, downward tilt, zoom angle, look-ahead distance, and steering gain value. The control goal was to keep the control point defined by the crop row edge at the horizontal control line on the image (specified by the look-ahead distance) in the vertical image centerline. When the control point went off the centerline, the wheels were steered towards the

control point in the vehicle reference frame by a proportional gain (a steering gain of 1.0 steered the wheels directly at the point). A steering angle dead band was used to reduce steering actuator chatter. Three degrees was chosen for the dead band though experimentation. The vehicle successfully navigated along straight line paths.

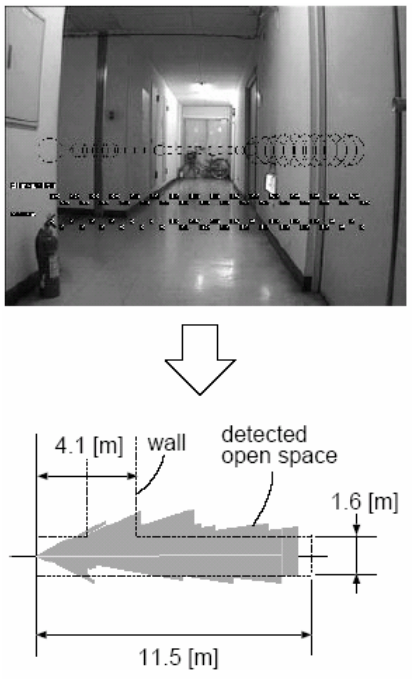


Figure 2-3. Free space from stereovision (Matsumoto et al., 2000). Upper: result of stereo matching (larger circles represent closer distances to the camera). Lower: detected free space in front of the robot.

Centroid of Path Row from Blob Analysis

Benson et al. (2003) demonstrated a vision guidance system for a combine harvester. The camera was mounted 3.2 m above the ground, directly over the center of the vehicle, 1.5 m ahead of the front axle, and with a downward pitch of 15 degrees. The mounting and field of view of the camera is shown in Figure 2-4. This configuration provided stronger contrast of the cut and uncut crop for better vehicle guidance than a lower mounted camera. A blob analysis was completed on the space between the crop

rows (Figure 2-5), and the vehicle was guided by the centroid of the center blob after verification by a fuzzy module. A PID controller was used for path following.

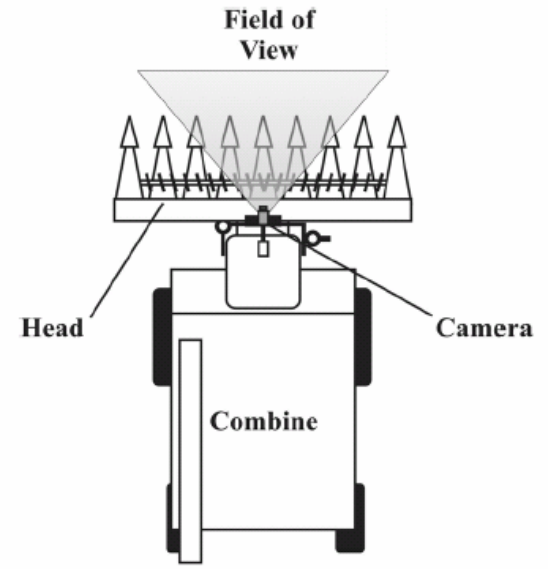
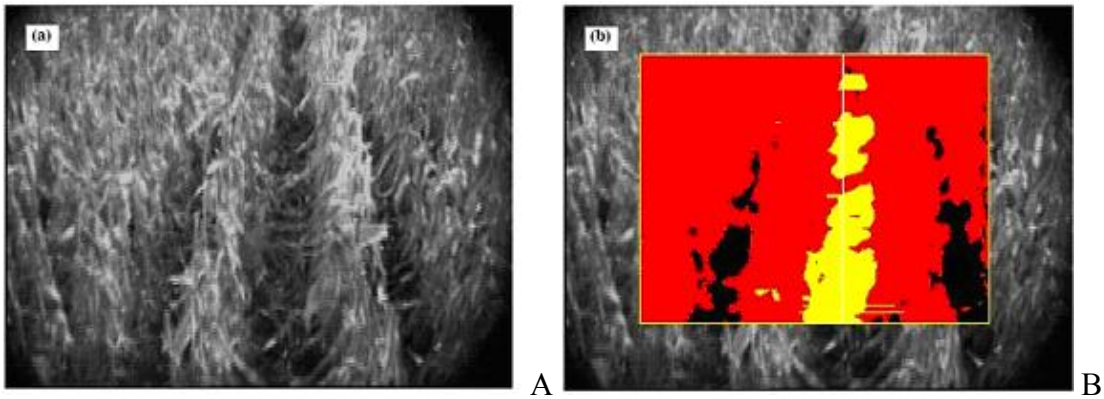


Figure 2-4. Monochrome camera located above the cab of the combine (Benson et al., 2003).



A B

Figure 2-5. Representative images from the cab mounted camera: A) unprocessed image; B) processed image (Benson et al., 2003).

Crop Line Boundary with Pure Pursuit

Ollis and Stentz (1997) developed a vision system for crop line tracking for an automated harvester cutting fields of alfalfa hay. The crop line was detected by running a best fit step function of scan lines in the color image and dividing the image into two regions: cut and uncut crops. An adaptive discriminant function was used to compensate

for changes in crop and soil color over time, and shadow compensation was executed to detect the crop line when shadows were present. The crop line boundary pixels were followed using a pure pursuit path follower.

The limitations of Ollis and Stentz (1997) are that only a single boundary between cut and uncut crop, as well as a single end of row line, can be detected. The row must also be generally vertical and down the center of the image to detect an end of row. The navigation setup of the row tracking system explained in Ollis and Stentz (1996) requires the camera to be mounted off to the side of the vehicle and directly over the crop line to keep the cut line in the center of the image.

Detecting Intersections

The task of detecting intersections and end of rows was explored by Matsumoto et al. (2000), Rasmussen (2003), Jochem et al. (1996), Hague et al. (1997), and Ollis and Stentz (1997).

Optical Flow

In Matsumoto et al. (2000), optical flow was used to detect junctions in a corridor setting by looking for drop-offs in the optical flow from the passing hallway walls. An omnidirectional image (covering a 360° view) around the robot was used to view the environment. Figure 2-6 demonstrates the detection of an intersection using optical flow in a sequence of omnidirectional views. In the figure, the features in the boxed region of the left image have not moved as much as the region in the right image. The left view is determined to be a hallway because features farther away do not shift in the image with the vehicle movement, leading to the conclusion that an intersection has been detected.

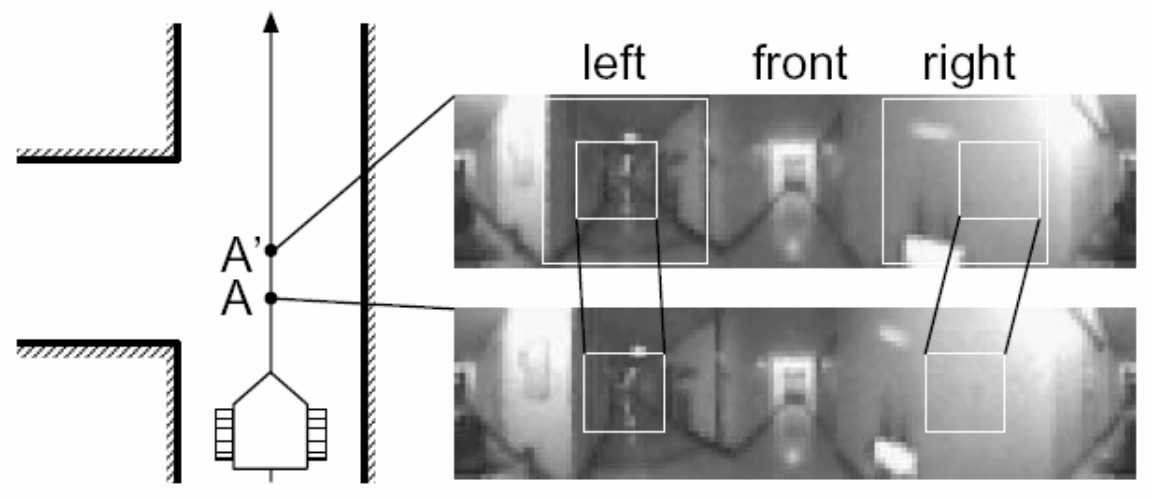


Figure 2-6. Optical flows obtained on both sides of the robot at an intersection (Matsumoto et al., 2000).

Support Vector Machine Modeling

Rasmussen (2003) used a set of three cameras as shown in Figure 2-7 to acquire a horizontal field of vision of 93°. The three separate images were stitched together and projected into a bird's-eye view, or rectified image, of the road with a planar ground assumption (Figure 2-8). The road was segmented from the image based on a road color model trained with a support vector machine, as shown in Figure 2-9. Three intersections types: cross, T, and 90-degree turns, keyed at a specified distance ahead of the vehicle, were modeled with a Support Vector Machine using training images of the intersections. Intersections appearing at the specified distance ahead were detected with an accuracy of 89.0% and classified from these models using a nearest neighbor approach with an accuracy of 97.1%.

Virtual Camera View Projection and Neural Networks

Jochem et al. (1996) used a pan-tilt camera with the ALVINN neural network road navigation system. Intersections were detected by analyzing virtual camera views directed a specified distance in front of the vehicle at various angles. Figure 2-10

demonstrates this technique. In Images 1 through 5, the virtual camera views directed at each branch angle are outlined in the image in white and shown in the bottom-left corner, as used in the neural network road classification. A road detected by the neural network at each virtual camera view was considered a branch of an intersection, and the intersection was classified based on the arrangement of branches. The intersection in the figure is determined to be a "T" intersection as shown in Image 6.

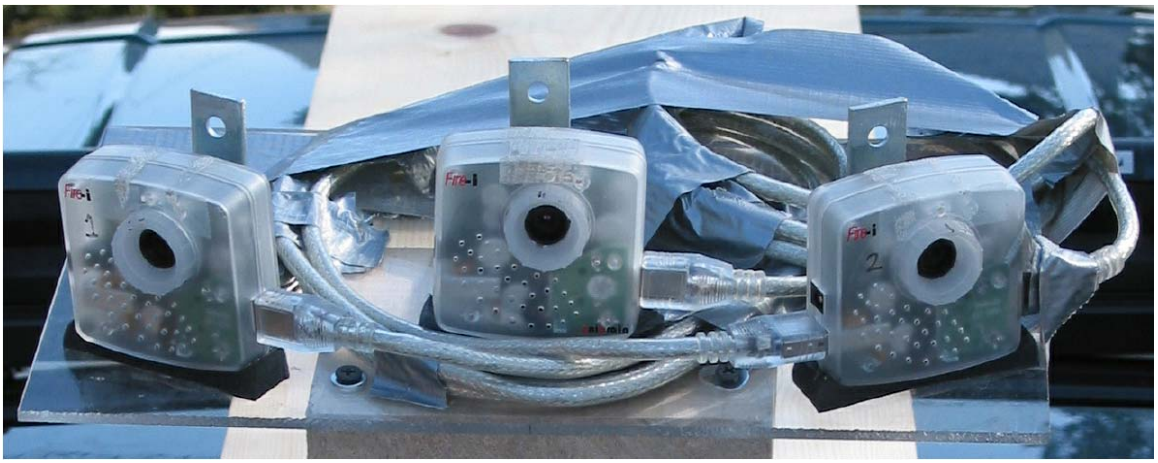


Figure 2-7. Polycamera arrangement (Rasmussen, 2003).

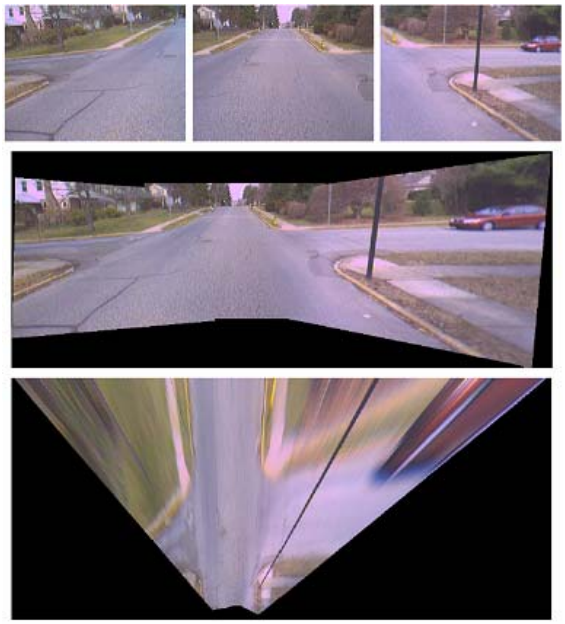
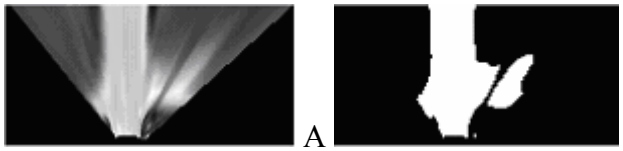


Figure 2-8. Polycamera road images, mosaic, and rectified mosaic (Rasmussen, 2003).



A B

Figure 2-9. Segmented road scene: A) Rectified road likelihood mosaic; B) Rectified segmentation mosaic (Rasmussen, 2003).

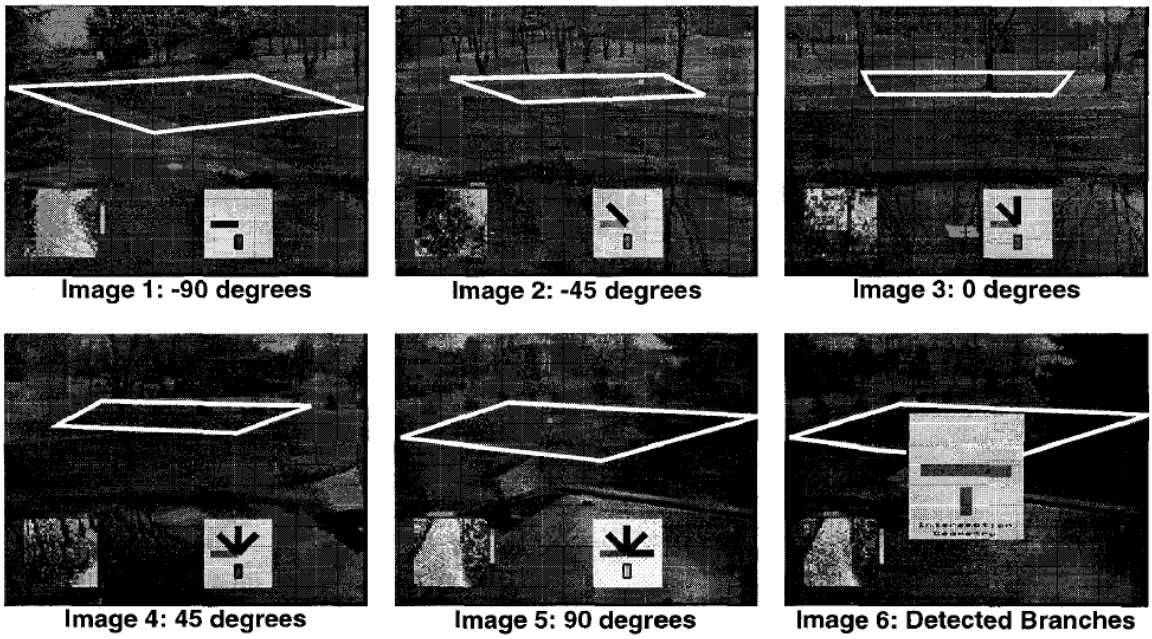


Figure 2-10. Searching for "T" intersection branches (Jochem et al., 1996).

End of Row Detection from Crop Absence

The robot in Hague et al. (1997) detected an end of a row by visually finding an absence of plants down a row that occurred at a distance greater than the estimated row length. Lines derived from the Hough Transform described the plant rows.

Ollis and Stentz (1997) detected end of rows by finding the image row best separating scan lines containing a crop line boundary and scan lines that don't. Figure 2-11 shows a processed image displaying the estimated crop line boundary and end of row. The detected end of row was verified by proving that 1) the end of row was within a specified distance from the vehicle, and 2) a series of processed images showed the

detected end of row approaching. Experiments with the navigation system proved successful in detecting end of rows.

Figure 2-11. Sample output from the crop line tracker (Ollis and Stentz, 1997).

Traversing Intersections

The task of traversing intersections was demonstrated in Jochem et al. (1996), Hague et al. (1997), and Fehr and Gerrish (1995).

Guidance from Projected Camera Views

In Jochem et al. (1996), the vehicle was guided though an intersection in two steps. First, the location and orientation of the intersection in relation to the vehicle was calculated. Figure 2-12 shows how the projection of two virtual camera views down an intersection branch were used to determine the orientation of the branch. The offset of the "Projected View" from the centerline was used to calculate this orientation. Second, a path for the vehicle to take was generated by calculating the arc that connects the current position of the vehicle and a point on the desired branch of the intersection (Figure 2-13).

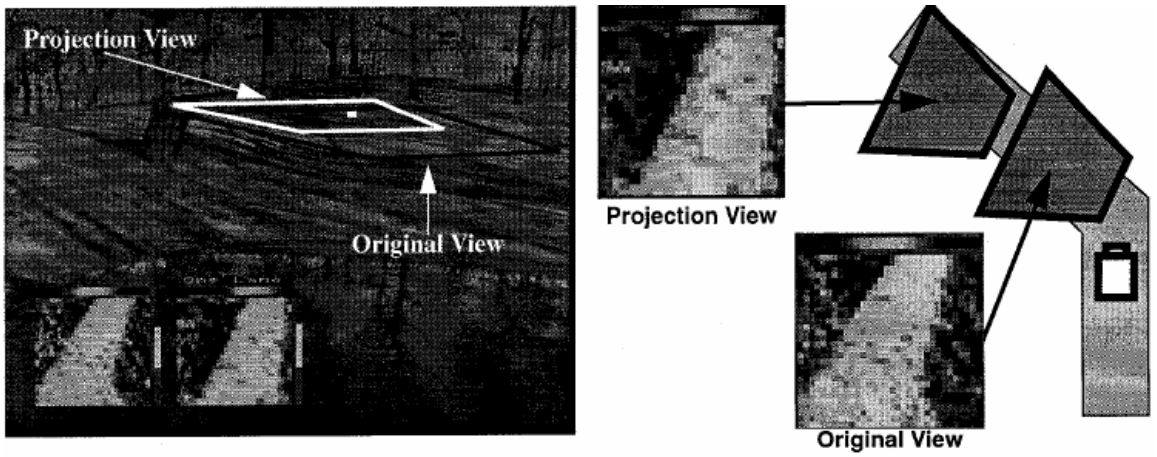


Figure 2-12. Orientation localization (Jochem, 1996).

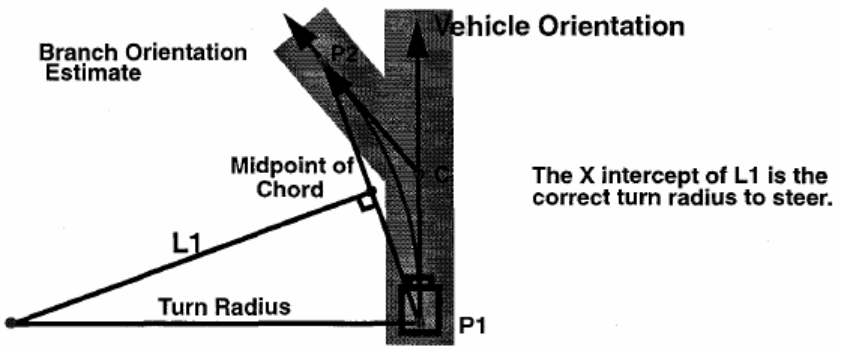


Figure 2-13. Traversal turn radius determination (Jochem, 1996).

Magnetic Compass, Plant Row Structure, and Dead Reckoning

The robot in Hague et al. (1997) responded at the end of a row by carrying out a planned path containing a 180-degree turn suitable for a constant row spacing of the crops. The vehicle position was controlled along the 180-degree path using estimates of the vehicle position and orientation angle calculated from a magnetic compass and visually observing the plant row structure extracted with the Hough Transform.

Fehr and Gerrish (1995) performed end of row maneuvers using pre-programmed dead reckoned three-point turns with position and orientation error within six inches and five degrees.

Sensors

Vehicle navigation has been explored using various sensors and combinations of sensors. The Demeter System, developed at Carnegie Mellon in Pilarski et al. (1999), utilized differential GPS, wheel encoders (dead reckoning), a gyro system, and a set of cameras for navigation. Rasmussen (2002) used a three-dimensional laser range-finder Schwartz SEO Ladar to gather depth information that could cover a view of 90 degrees horizontally and 15 degrees vertically. Stereo vision used in Matsumoto et al. (2000), multi-camera configurations as in Rasmussen (2003), wide-angle lenses, and controllable pan-tilt mounts used in Jochem et al. (1996) provided more information about the environment than a single camera, but also created additional costs, hardware complexity, and calibration needs. GPS can be unreliable when satellites become occluded by trees and other environmental landmarks and require a detailed map with waypoint coordinates to follow. Wheel encoders accumulate error, especially with soft and uneven terrain, which causes the wheels to slip. Gyro systems also drift over time and can be expensive.

The autonomous greenhouse sprayer developed by Singh (2004) was guided through simulated greenhouse alleys using ultrasonic sensors, ladar, and machine vision.

Ultrasonic Range Sensor Navigation

Singh (2004) used two Migatron RPS-401A ultrasonic range sensors to navigate the vehicle along the center of a straight path. The sensors were mounted on the front, pointing to the right and left. Path error was calculated by subtracting the right and left distances read by the sensors. Testing using ultrasonic sensors is shown in Figure 2-14. Results from the navigation system are reported in Singh (2004).

Figure 2-14. Testing with ultrasonic sensors (Singh, 2004).

The benefits of ultrasonic range sensors are low-cost and accuracy. The downfalls are that their field of view is limited and the path boundaries must by well aligned within the sensors view. The ultrasonic sensors used also required well-defined surfaces with the reflecting surface tilted no more than 8 degrees.

Ladar Navigation

Sing and Subramanian (2004) used a Sick LMS 200 laser radar to navigate the vehicle down the center of straight, moderately curved, and hard curved paths. The ladar used measured distance at angles from 0 to 180 degrees. The first and last 15 degrees were used to determine the left and right distances to the path edge. The path error was computed by subtracting the left and right edge distances. A corridor environment was used to simulate greenhouse rows during experimentation, as shown in Figure 2-15.

Ladar provides accurate distance measurements around the vehicle. However, ladar has limitations in its field of view, the types of objects it can detect (objects must reflect the laser pulses from the Ladar back to the Ladar system), and are high in cost, as explained in Fernandez and Casals (1997).

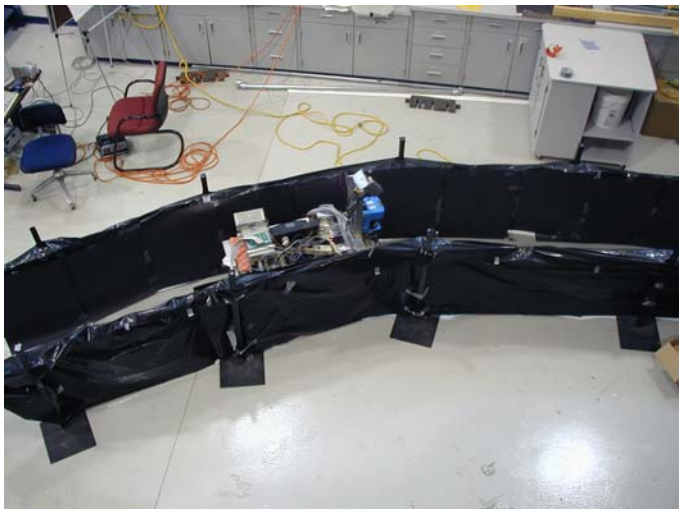


Figure 2-15. Moderately curved path setup for ladar and machine vision testing.

Machine Vision Navigation

A Sony FCB-EX7805 CCD camera with an Integral Technologies Flashbus MV Pro frame grabber was used by Singh and Subramanian (2004) to guide the vehicle down the path center in the machine vision approach. The camera was mounted above the front of the vehicle to acquire a full view of the path. Through image processing, the center of the path was found and an offset error was calculated. Experimentation was performed using the same setup as the ladar experimentation, as shown in Figure 2-15.

Machine vision is beneficial due to the low-cost of CCD cameras and the large amount of information that can be gathered from the path in front of the vehicle to use for navigation. Machine vision is limited to the field of view of the camera in that it cannot see the sides of the vehicle directly and must control the robot from a specified look-ahead distance further down the path.

Depth Perception

Active sensors, such as ultrasonic and laser radar, give depth information directly. To gather depth information from CCD cameras, other techniques must be applied.

The most common technique to gather depth from a CCD camera is stereovision. Stereovision requires two or more cameras focused on the same scene. Matsumoto et al. (2000) used stereovision to determine the distance to walls during vehicle navigation. If the assumption is made that the ground plane is flat, 3D locations of pixels and features on the path can be approximated from a single image. Fernandez and Casals (1997) used this assumption to locate an image pixel in the vehicle reference frame using a set of projection equations and found the mean error to be 0.5% of the world point distance from this frame (10 cm error for a pixel projected to be 20 meters away). Morimoto et al. (2002) also used the ground plane assumption to estimate distance from the vehicle to an obstacle using coordinate transformation between the camera and vehicle reference frames with an error within 0.3 meters. The relationship between path lines in the vehicle plane and the path lines in the image plane using a pinhole camera model are shown in Figure 2-16.

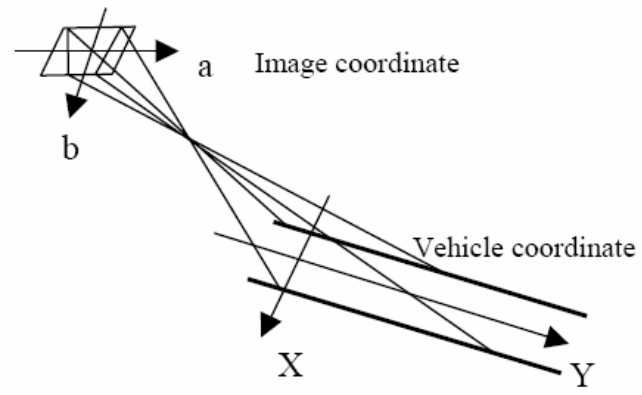


Figure 2-16. Coordinate transformation between image plane and vehicle coordinate.

Other methods have been explored to determine distance from a single camera. To name a few, Lavest et al. (1993) and Lavest et al. (1997) investigated methods for 3D reconstruction by zooming. Testing showed accurate results, but only for objects up to two meters away. Additionally, an extensive calibration procedure was necessary, and

the zoom measurement must be calculated on a stationary vehicle. Baba et al. (2002) gathered depth information from defocus and zooming. Unfortunately, this technique was restrained to sharp, well-defined edges in uncomplicated scenes, and also must be performed on a stationary vehicle.

Dai et al. (1995) demonstrated a technique of determining the depth of an object from a moving vehicle by measuring the changes in the appearances of the object in subsequent images and knowledge of the vehicle motion. The two variances looked at were:

1. Increasing distance between two edges of the object
2. Increasing area of the object

This technique proved to be accurate with the use of a Kalman Filter. However, this method was limited to objects with edges the same distance to the camera and objects in the center of the image plane. Also, the forward movement of the vehicle must be straight and perpendicular to the object at all times during the image sequence, and reliable velocity information was required for the calculations.

Kundar and Raviv (1999) described a system to measure the relative change in range of an object by measuring the change in its texture. This requires the use of a visually fixating, fixed-focus, monocular camera moving towards the objects. Also, an accurate tracking of vehicle forward motion was needed. However, this technique provided no exact distance information, and could only tell when an object was approaching the fixed focal range set on the camera.

Structure from motion can also used to gather depth information and create a model of the environment (Pollefeys, 2004). This technique requires precise camera calibration and is computationally expensive.

Path Modeling

Several constraints can be made about the environment to allow the development of a navigation model from image data. Fernandez and Casals (1997) described a set of constraints for ill-structured roads (dirt roads and mountain ways):

1. They have a locally constant direction.
2. Their left and right boundaries are parallel or almost parallel.
3. They have a locally constant width.
4. The materials that define their surface are small and spread over large areas.
5. Their slope is locally constant.

CHAPTER 3
MATERIALS AND METHODS

This chapter covers the general control architecture for the robotic greenhouse sprayer and techniques developed for path following, intersection detection, and intersection navigation. The following topics are covered:

- Control Architecture- The general overview is given of the hardware used to guide the vehicle.
- Sensors- Selection of a CCD camera as the primary sensor for vehicle navigation is discussed.
- Camera Model- A model is developed to relate pixels in the camera image to 3D coordinates in the vehicle coordinate system. Intrinsic and extrinsic parameters used to define the camera model are discussed. Procedures to 1) transform 3D points in the vehicle system to pixels in the image plane and 2) transform pixels to 3D points in the vehicle system are described.
- Image Processing- Steps taken to find the path in the image are described.
- Path Analysis and Intersection Detection- Techniques to detect the path center, path edges, and intersection in the image are discussed.
- Path Following- The control strategy to follow the path center and path edges is covered.
- Visual Odometer- A visual odometer created to track vehicle translation and orientation by tracking ground features is discussed.
- Intersection Navigation- Algorithms developed to command the vehicle to 1) turn at an intersection and 2) drive straight through an intersection are described.

Control Architecture

Figure 3-1 shows the general control architecture used to guide the robotic greenhouse sprayer. A camera provided information about the scene. The PC acquired

images from the camera, processed the image, and output two separate control signals ranging from 0 to 2.5 Volts to drive the left and right motor. Amplifiers scaled the control signals to 24 Volts. Finally, DC motor controllers drove the motors. The work developed in this thesis focuses on the vehicle navigation control strategy for the PC.

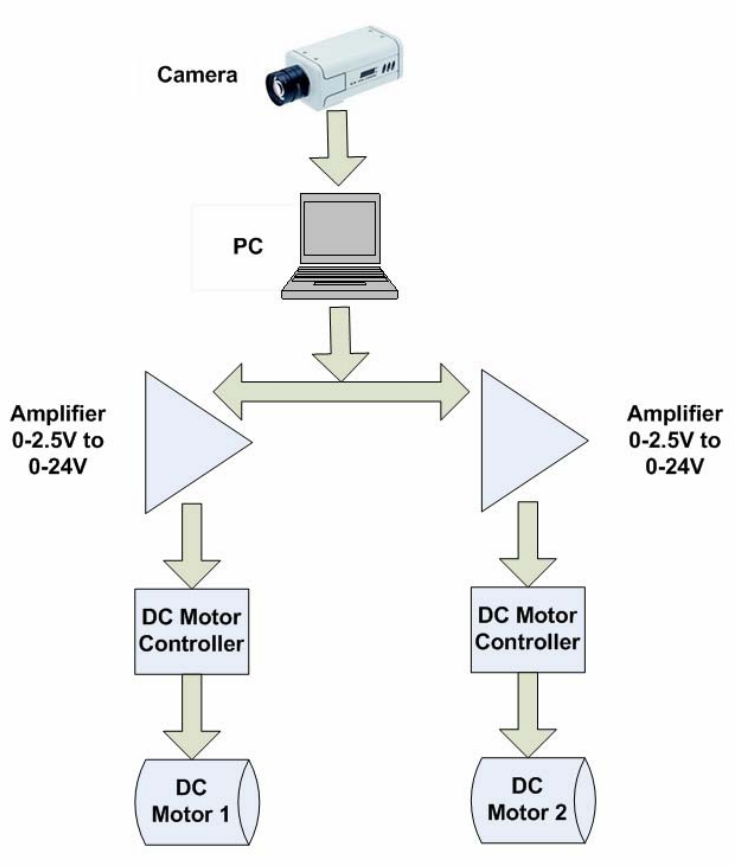


Figure 3-1. Control architecture for the robotic greenhouse sprayer.

Sensors

A Sony FCB-EX7805 CCD camera was utilized as the primary sensor for vehicle navigation. A single camera was chosen as opposed to a multiple camera setup or stereovision to reduce costs that come with additional equipment and reduce complexity associated with calibrating multiple cameras. An Integral Technologies Flashbus MV Pro frame grabber was used to capture 640x480 pixel color images from the camera.

Camera Model

The autonomous greenhouse sprayer was equipped with a Sony FCB-EX7805 CCD camera and a Flashbus MV Pro frame grabber from Integral Technologies.

Sets of intrinsic and extrinsic camera parameters were found to describe the camera. Intrinsic parameters account for focal length, principal point, skew, and lens distortion. Extrinsic parameters account for the rotation and translation of the camera coordinate system relative to the vehicle coordinate system. Using the intrinsic and extrinsic camera parameters, ground points in the robot coordinate system can be transformed into pixel coordinates and vice-versa.

Intrinsic Parameters

The intrinsic camera parameters relate 3D points in the camera coordinate system to pixel coordinates. These parameters were determined using the Camera Calibration Toolbox for Matlab developed by Bouguet (2004). Figure 3-2 shows the relationship between the vehicle and pixel coordinate systems.

The calibration optimization results, including pixel skew and sixth-order radial distortion, are shown below:

Focal Length: fc = [869.55; 867.78]

Principal Point: cc = [318.85; 249.62]

Skew: alpha_c = 0.0023070

Distortion: kc = [-0.33118; 1.0758; 0.0011577; 0.00074029; -3.0793]

Extrinsic Parameters

The extrinsic camera parameters relate 3D points in vehicle coordinate system to the camera coordinate system. The vehicle and camera coordinate systems are displayed in Figure 3-3.

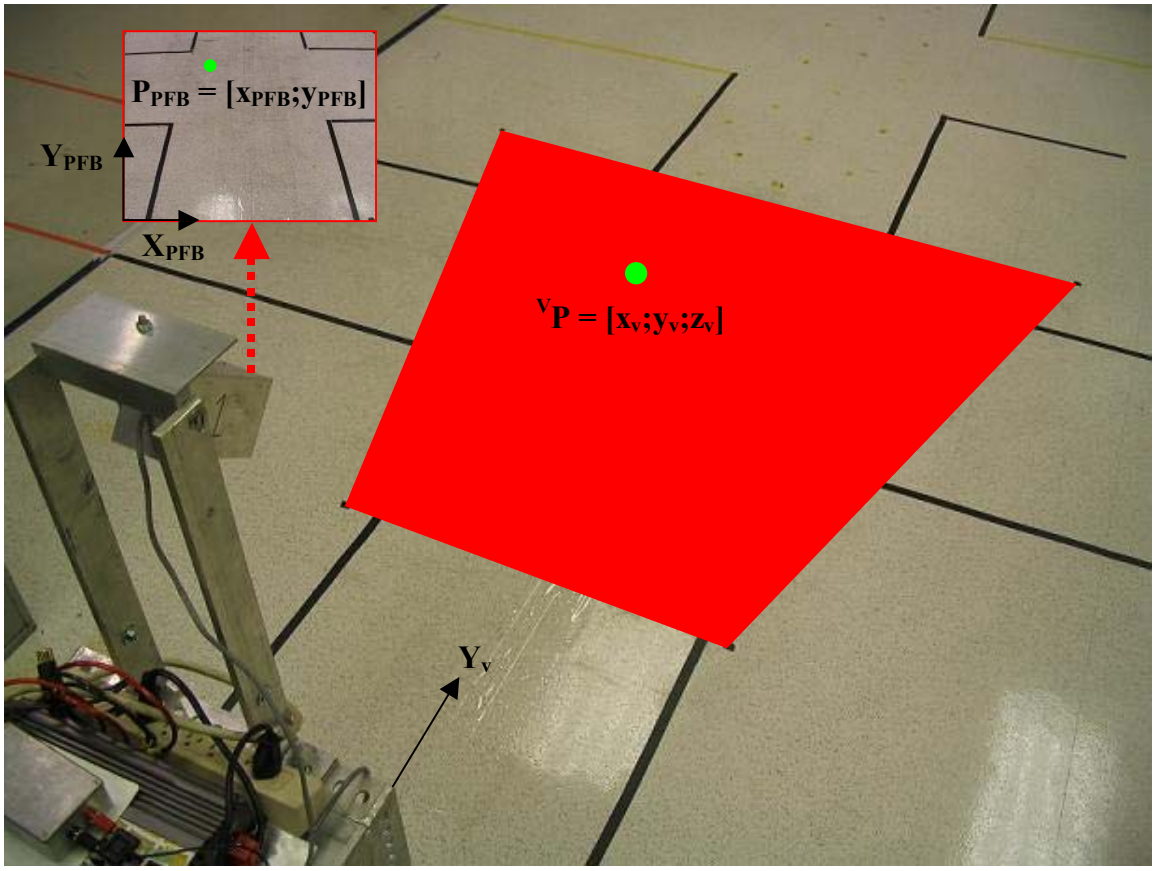


Figure 3-2. Relationship between the Vehicle Coordinate System [$x_V;y_V;z_V$] and the Flashbus Pixel Coordinate System [$x_{PFB};y_{PFB}$]. The field of view of the camera is highlighted in red. Point P represented by the green circle is shown both in the vehicle and Flashbus systems.

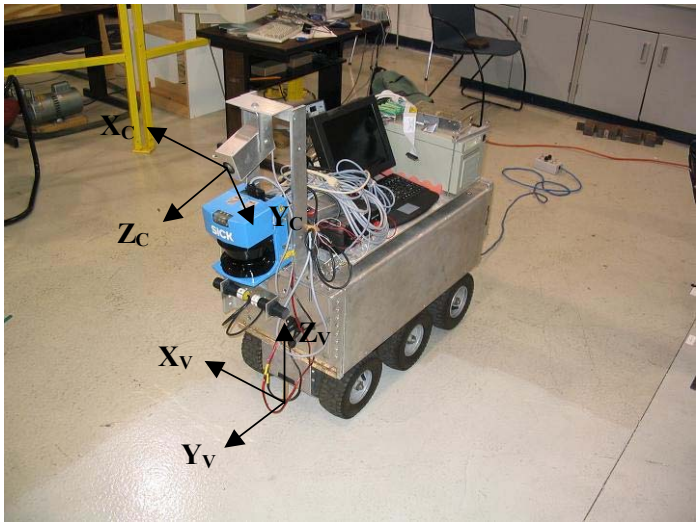


Figure 3-3. Camera (C) and vehicle (V) coordinate systems on the autonomous greenhouse sprayer.

The extrinsic parameters consist of a 3x3 rotation matrix, R_C, and a 3x1 translation matrix, T_C

$$R_C = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \qquad T_C = \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

The rotation matrix indicates the orientation of the camera coordinate system in relation to the vehicle reference frame. The translation matrix indicates the position of the camera coordinate system origin in relation to the vehicle reference frame. These two matrices are combined into a single Transformation matrix, ${}^{C}T_{V}$

$${}^{C}T_{V} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_X \\ R_{21} & R_{22} & R_{23} & T_Y \\ R_{31} & R_{32} & R_{33} & T_Z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

${}^{C}T_{V}$ transforms a point in the vehicle coordinate system, ${}^{V}P = [{}^{V}x;\ {}^{V}y;\ {}^{V}z]$ to the camera coordinate system, ${}^{C}P = [{}^{C}x;\ {}^{C}y;\ {}^{C}z]$, through the following equation

$${}^{C}P = {}^{C}T_{V} \cdot {}^{V}P$$

To calculate ${}^{C}T_{V,}$ intermediate coordinate systems consisting of simple rotation and translations matrices wre used. Figure 3-4 and Figure 3-5 show these intermediate coordinate systems and measurements between them. The values of the parameters used are shown in Table 3-1.

A transformation matrix was calculated to move between each coordinate system. To translate a distance T_x along the x-axis, T_y along the y-axis, or T_z along the z-axis of coordinate system “A” to get to coordinate system “B”, the following transformation matrix used is

$$ {}^{B}T_{A} = \begin{bmatrix} 1 & 0 & 0 & T_X \\ 0 & 1 & 0 & T_Y \\ 0 & 0 & 1 & T_Z \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

To rotate an angle α about the x-axis of coordinate system "A" to get to coordinate system "B", the transformation matrix used is

$$ {}^{B}T_{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & \sin(\alpha) & 0 \\ 0 & -\sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

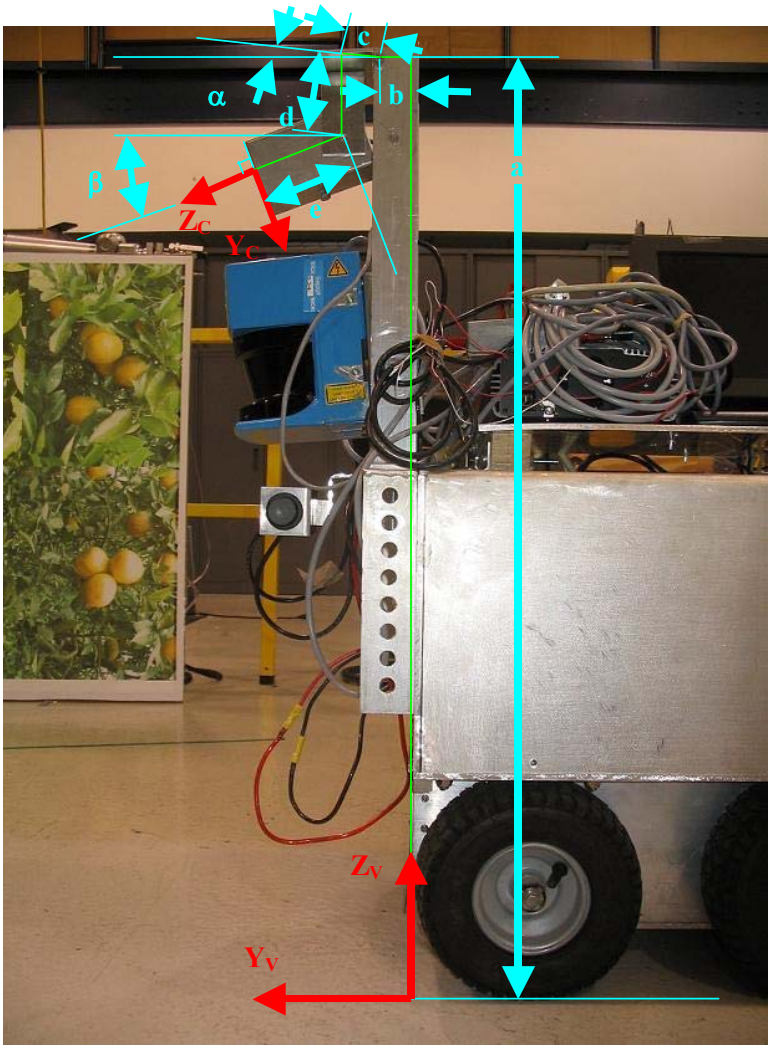


Figure 3-4. Measurements from vehicle to camera coordinate system.

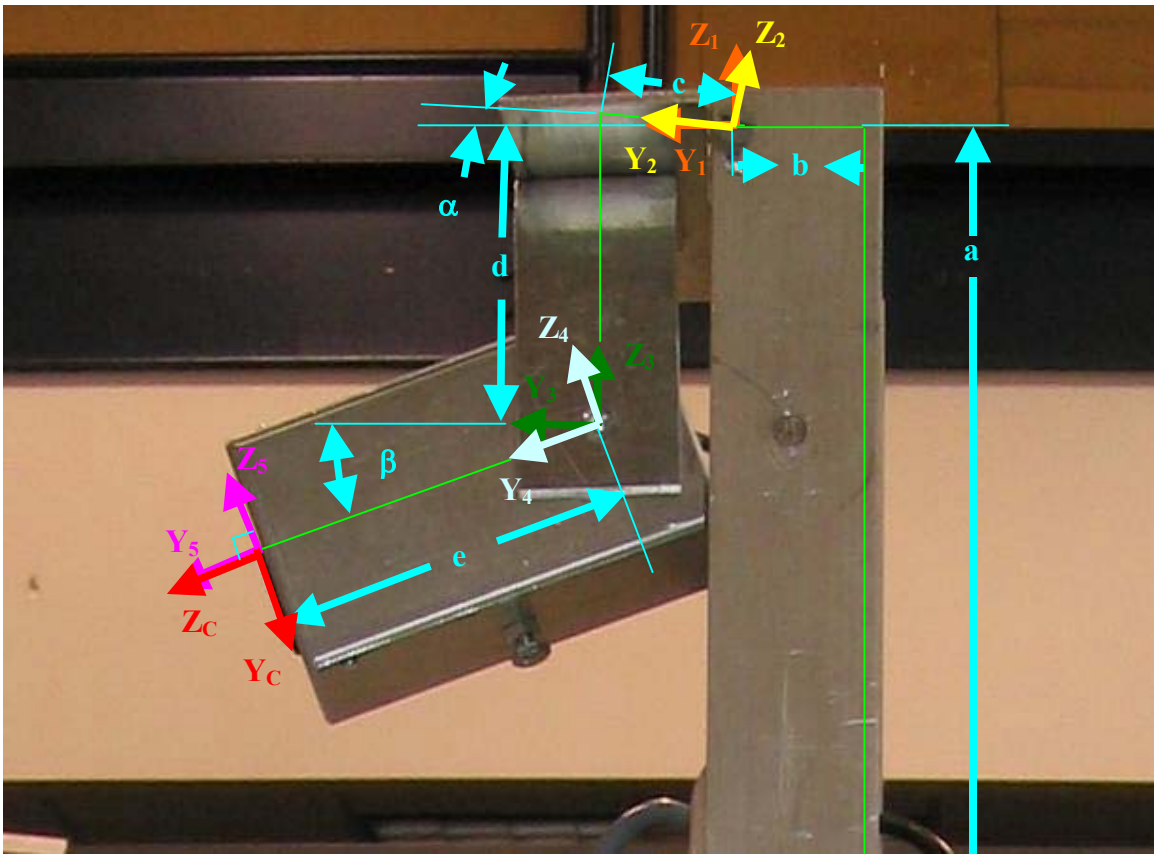


Figure 3-5. Intermediate coordinate systems.

Table 3-1. Parameters relating coordinate systems.

Parameter	**Value**
a	37.875 inches
b	1.625 inches
c	1.875 inches
d	2.125 inches
e	4.25 inches
α	0°
β	40°

Using the appropriate translation and rotation transformation matrices to move to each of the coordinate systems by the given measurements lead to the following matrices

$$ {}^{1}T_{V} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & a \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^{2}T_{1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & \sin(\alpha) & 0 \\ 0 & -\sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{3}T_{2} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & c \\ 0 & 0 & 1 & -d \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^{4}T_{3} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\beta) & \sin(\beta) & 0 \\ 0 & -\sin(\beta) & \cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{5}T_{4} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & e \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^{C}T_{5} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(-90°) & \sin(-90°) & 0 \\ 0 & -\sin(-90°) & \cos(-90°) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

${}^{C}T_{V}$ is found by multiplying each of the transformations matrices

$$ {}^{C}T_{V} = {}^{C}T_{5} \cdot {}^{5}T_{4} \cdot {}^{4}T_{3} \cdot {}^{3}T_{2} \cdot {}^{2}T_{1} \cdot {}^{1}T_{V} $$

Using the parameters defined in Table 3-1, ${}^{C}T_{V}$ was calculated

$$ {}^{C}T_{V} = \begin{bmatrix} 1.00 & 0 & 0 & 0 \\ 0 & -0.643 & 0.766 & 172 \\ 0 & -0.766 & -0.643 & 839 \\ 0 & 0 & 0 & 1.00 \end{bmatrix} $$

Using homogenous coordinates, ${}^{V}T_{C}$ transforms a point in the camera coordinate system, ${}^{C}P = [{}^{C}x;\ {}^{C}y;\ {}^{C}z;\ 1]$, to the vehicle coordinate system, ${}^{V}P = [{}^{V}x;\ {}^{V}y;\ {}^{V}z;\ 1]$, using the equation

$$ {}^{V}P = {}^{V}T_{C} \cdot {}^{C}P $$

${}^{V}T_{C}$ is found by taking the inverse of ${}^{C}T_{V}$

$$ {}^{V}T_{C} = {}^{C}T_{V}{}^{-1} $$

Using the parameters defined in Table 3-1, ${}^{V}T_{C}$ was calculated

$$ {}^{V}T_{C} = \begin{bmatrix} 1.00 & 0 & 0 & 0 \\ 0 & -0.643 & -0.766 & 172 \\ 0 & 0.766 & -0.643 & 408 \\ 0 & 0 & 0 & 1.00 \end{bmatrix} $$

Transforming 3D Points in Vehicle System to Pixels in Flashbus Pixel Plane

Given a point in the vehicle coordinate system, ${}^{V}P = [{}^{V}x; {}^{V}y; {}^{V}z]$, the Flashbus pixel coordinate, $P_{PFB} = [x_{PFB}; y_{PFB}]$, was calculated using the following steps:

1. Find point in camera coordinate system, ${}^{C}P$, using homogenous coordinates

$$ {}^{C}P = {}^{C}T_{V} \cdot {}^{V}P $$

where ${}^{C}T_{V}$ is the transformation matrix defined in the camera extrinsic parameters.

2. Given ${}^{C}P = [{}^{C}x; {}^{C}y; {}^{C}z]$, find normalized (pinhole) image projection of point on image plane, P_n

$$ P_n = \begin{bmatrix} {}^{C}x/{}^{C}z \\ {}^{C}y/{}^{C}z \\ 1 \end{bmatrix} $$

3. Given $P_n = [x_n; y_n; 1]$, adjust for lens distortion

$$ P_d = (1 + kc[1] \cdot r^2 + kc[2] \cdot r^4 + kc[5] \cdot r^6) \cdot P_n + dx $$

where dx is the tangential distortion vector defined from the intrinsic camera parameters

$$ dx = \begin{bmatrix} 2 \cdot kc[3] \cdot x_n \cdot y_n + kc[4] \cdot (r^2 + 2 \cdot {x_n}^2) \\ kc[3] \cdot (r^2 + 2 \cdot {y_n}^2) + 2 \cdot kc[4] \cdot x_n \cdot y_n \\ 1 \end{bmatrix} $$

4. Given $P_d = [x_d; y_d; 1]$, find for pixel coordinates

$$ P_{Pixel} = \begin{bmatrix} x_{Pixel} \\ y_{Pixel} \\ 1 \end{bmatrix} = KK \cdot P_d $$

where KK is the camera matrix defined from the intrinsic camera parameters

$$ KK = \begin{bmatrix} fc[1] & alpha\_c \cdot fc[1] & cc[1] \\ 0 & fc[2] & cc[2] \\ 0 & 0 & 1 \end{bmatrix} $$

5. For a 640 x 480 image, transform pixel coordinates (with origin in upper left corner of image) to Flashbus pixel coordinates (with origin in lower left corner of image)

$$P_{PFB} = \begin{bmatrix} x_{PFB} \\ y_{PFB} \end{bmatrix} = \begin{bmatrix} x_{Pixel} \\ 480 - y_{Pixel} \end{bmatrix}$$

Refer to Bouguet (2004) for more discussion on steps 2 through 5.

Transforming Pixels in Flashbus Pixel Plane to 3D Points in Vehicle System (only for Ground Pixels)

Given a pixel in the Flashbus pixel coordinate, $P_{PFB} = [x_{PFB}; y_{PFB}]$, a point in the vehicle coordinate system, ${}^{V}P = [{}^{V}x; {}^{V}y; {}^{V}z]$, is calculated using the following steps:

1. For a 640 x 480 image, transform from Flashbus pixel coordinates (with origin in lower left corner of image) to general pixel coordinates (with origin in upper left corner of image)

$$P_{Pixel} = \begin{bmatrix} x_{Pixel} \\ y_{Pixel} \end{bmatrix} = \begin{bmatrix} x_{PFB} \\ 480 - y_{PFB} \end{bmatrix}$$

2. Transform from pixel to distortion coordinate system

$$P_d = \begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} = KK^{-1} \cdot \begin{bmatrix} x_{Pixel} \\ y_{Pixel} \\ 1 \end{bmatrix}$$

where KK is the camera matrix defined from the intrinsic camera parameters

$$KK = \begin{bmatrix} fc[1] & alpha\_c \cdot fc[1] & cc[1] \\ 0 & fc[2] & cc[2] \\ 0 & 0 & 1 \end{bmatrix}$$

3. Convert from distortion to normal coordinate system based on iterative algorithm in "normalize.m" and "comp_distortion_oulu.m" in Camera Calibration Toolbox developed by Bouguet (2004).

Initialize P_n to be P_d

$$P_n = P_d$$

Perform iteration

for i = 1 to n

{

$$dx = \begin{bmatrix} 2 \cdot kc[3] \cdot x_n \cdot y_n + kc[4] \cdot (r^2 + 2 \cdot x_n{}^2) \\ kc[3] \cdot (r^2 + 2 \cdot y_n{}^2) + 2 \cdot kc[4] \cdot x_n \cdot y_n \end{bmatrix}$$

$$x\_new = \frac{x_d - dx[1]}{1 + kc[1] \cdot r^2 + kc[2] \cdot r^4 + kc[5] \cdot r^6}$$

$$y\_new = \frac{y_d - dy[1]}{1 + kc[1] \cdot r^2 + kc[2] \cdot r^4 + kc[5] \cdot r^6}$$

$$P_n = \begin{bmatrix} x\_new \\ y\_new \\ 1 \end{bmatrix}$$

}

where n is number of iterations desired for P_n to converge. Through experimentation, five iterations were found adequate (n = 5) for negligible error.

4. Given $P_n = [x_n; y_n; 1]$, convert from normal to vehicle coordinate system, ${}^{V}P = [{}^{V}x; {}^{V}y; {}^{V}z]$. These coordinates are solved by using the equation relating a point in the vehicle coordinate system, ${}^{V}P$, and a point in the camera coordinate system, ${}^{C}P$, with the transformation matrix, ${}^{C}T_V$, defined in the camera extrinsic parameters

$${}^{C}P = {}^{C}T_V \cdot {}^{V}P$$

Expanding the matrices

$$\begin{bmatrix} {}^{C}x \\ {}^{C}y \\ {}^{C}z \\ 1 \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} \\ T_{21} & T_{22} & T_{23} & T_{24} \\ T_{31} & T_{32} & T_{33} & T_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} {}^{V}x \\ {}^{V}y \\ {}^{V}z \\ 1 \end{bmatrix}$$

Next, an assumption is made that the pixel lies on the ground plane (${}^{V}z = 0$)

$$\begin{bmatrix} {}^{C}x \\ {}^{C}y \\ {}^{C}z \\ 1 \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} \\ T_{21} & T_{22} & T_{23} & T_{24} \\ T_{31} & T_{32} & T_{33} & T_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} {}^{V}x \\ {}^{V}y \\ 0 \\ 1 \end{bmatrix}$$

Multiplying the right side of the equation gives

$$\begin{bmatrix} {}^{C}x \\ {}^{C}y \\ {}^{C}z \\ 1 \end{bmatrix} = \begin{bmatrix} T_{11} \cdot {}^{V}x + T_{12} \cdot {}^{V}y + T_{14} \\ T_{21} \cdot {}^{V}x + T_{22} \cdot {}^{V}y + T_{24} \\ T_{31} \cdot {}^{V}x + T_{32} \cdot {}^{V}y + T_{34} \\ 1 \end{bmatrix}$$

Solving for the normalized coordinates, $P_n = [x_n; y_n; 1]$

$$\begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{C}x/{}^{C}z \\ {}^{C}y/{}^{C}z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x_n \\ y_n \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} \dfrac{T_{11} \cdot {}^{V}x + T_{12} \cdot {}^{V}y + T_{14}}{T_{31} \cdot {}^{V}x + T_{32} \cdot {}^{V}y + T_{34}} \\ \dfrac{T_{21} \cdot {}^{V}x + T_{22} \cdot {}^{V}y + T_{24}}{T_{31} \cdot {}^{V}x + T_{32} \cdot {}^{V}y + T_{34}} \\ 1 \\ 1 \end{bmatrix}$$

Solving for ${}^{V}x$ and ${}^{V}y$ in terms of ${}^{n}x$ and ${}^{n}y$ gives the following equations for the x- and y-coordinates of the pixel in the vehicle system

$${}^{V}x = \frac{\left(T_{24} - y_n \cdot T_{34}\right) \cdot \left(x_n \cdot T_{31} - T_{11}\right) - \left(T_{14} - x_n \cdot T_{34}\right) \cdot \left(y_n \cdot T_{31} - T_{21}\right)}{\left(x_n \cdot T_{31} - T_{11}\right) \cdot \left(y_n \cdot T_{32} - T_{22}\right) - \left(x_n \cdot T_{32} - T_{12}\right) \cdot \left(y_n \cdot T_{31} - T_{21}\right)}$$

$${}^{V}y = \frac{\left(T_{14} - x_n \cdot T_{34}\right) - {}^{V}y \cdot \left(x_n \cdot T_{32} - T_{12}\right)}{\left(x_n \cdot T_{31} - T_{11}\right)}$$

The z-coordinate is know from the assumption that the pixel lies on the ground plane, ${}^{V}z = 0$. The pixel has now been transformed to a point in the vehicle coordinate system given by

$$ {}^{V}P = \begin{bmatrix} {}^{V}x \\ {}^{V}y \\ {}^{V}z \end{bmatrix} $$

Image Processing

Image processing is conducted on each image taken from the camera to determine where the path was in front of the vehicle. The following steps are taken during the image processing procedure:

1. Color threshold
2. Horizontal clean
3. Vertical clean
4. Extract largest path object
5. Keep nonpath objects not surrounded by path
6. Save largest horizontal path segments
7. Sample window to determine new threshold values

These steps are explained and demonstrated below using the sample greenhouse image shown in Figure 3-6.

Step 1: Color Threshold

Threshold values are based upon a statistical model of path pixels determined from analyzing the path found in the previous image. The threshold values consist of upper and lower limits for red, green, and blue pixel intensities on a scale of 0 to 255. A pixel is classified as "path" if each of its red, green, and blue components is within these limits. If either of the red, green, or blue components does not fall within their limits, the pixel is

classified as “nonpath.” The result of thresholding the image from Figure 3-6 using lower/upper limits for red, green, and blue of 67/138, 58/123, and 28/91 is shown in Figure 3-7. Pixels classified as “path” are shown as white, and pixels classified as “nonpath” are shown as black.

Figure 3-6. Sample image of greenhouse.

Figure 3-7. Thresheld image.

Step 2: Horizontal Clean

Each row of the segmented image from Step 1 is analyzed to remove small segments and noise. Starting at column 0 in the image, chains of connected pixels of the same class (“path” or “nonpath”) are recorded. If the length of the chain is equal to or larger than a specified pixel length, those pixels are not changed. If the length of the chain is smaller than the specified length, all pixels of that chain are changed to the class

of the pixel before it. The result of running the horizontal clean on image Figure 3-7 using a minimum horizontal chain length of 10 is shown in Figure 3-8.

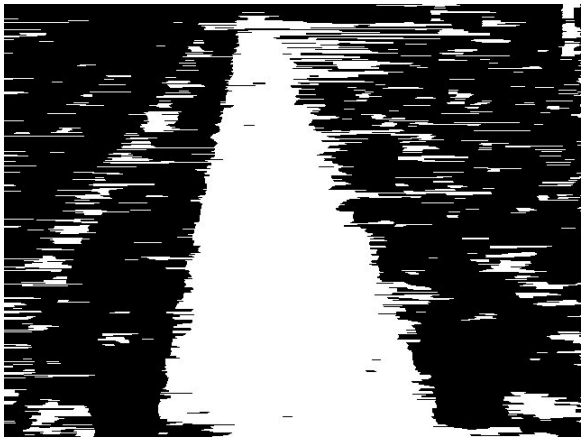


Figure 3-8. Horizontal cleaned image.

Step 3: Vertical Clean

Each column of the image from Step 2 is analyzed to remove small segments and noise. Starting at row 0 in the image, chains of connected pixels of the same class ("path" or "nonpath") are recorded. If the length of the chain is equal to or larger than a specified pixel length, those pixels are not changed. If the length of the chain is smaller than the specified length, all pixels of that chain are changed to the class of the pixel before it. The result of running the vertical clean on image Figure 3-8 using a minimum vertical chain length of 10 is shown in Figure 3-9.

Figure 3-9. Vertical cleaned image.

Step 4: Extract Largest Path Object

After the thresheld image is cleaned of small segments, the image is searched for path objects. Path objects are found by looking at "seed" pixels spaced out at specified row and column intervals on the image. If a seed pixel is found to be a path pixel and has not yet been labeled as a path object, it is labeled as a new object. All path pixels connected to that pixel (above, below, left, and right), as well as path pixels connected to these and so on, are also labeled as members of the object belonging to the seed pixel.

Path objects found in Figure 3-9 by placing seed pixels every 20 rows and 20 columns are displayed in Figure 3-10. A total of eight path objects were found. The red circles indicate the seed pixels that found the objects. Each object found is painted a different shade of blue from light to dark. Note that some path objects represented by white were not found. This is because seed pixels were not positioned on these objects during the search. This is not a problem since only the main path needs to be found, and it is assumed that this path will be large enough where at least one seed pixel will be positioned on it.

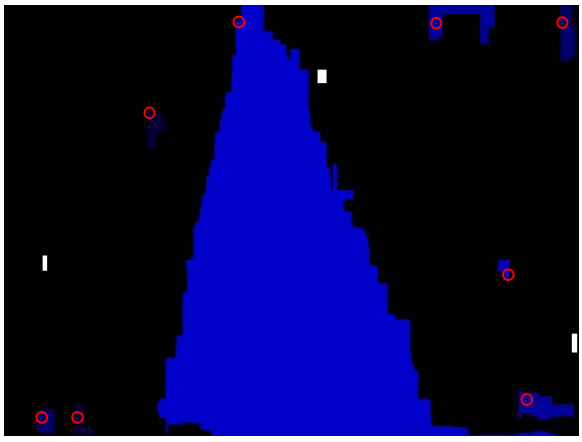


Figure 3-10. Path objects.

Only the path object with the largest pixel area is saved. The rest of the image surrounding the largest path object is classified as “nonpath”. Based on the path objects found in Figure 3-10, the resulting image is shown in Figure 3-11.

Figure 3-11. Extracted largest path object.

Step 5: Keep Nonpath Objects Not Surrounded by Path

A similar process as performed in Step 4 is used to remove nonessential nonpath objects. In this step, the image is searched for nonpath objects. Nonpath objects found in Figure 3-11 by placing seed pixels every 10 rows and 10 columns are displayed in Figure 3-12. A total of two nonpath objects were found. The red circles indicate the seed pixels that found the objects. Each object found is painted a different shade of blue from light to dark.

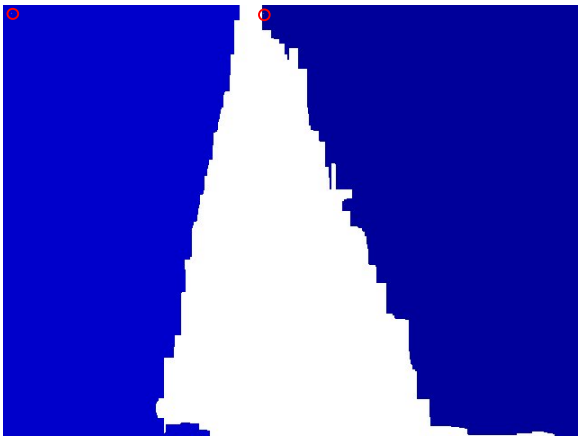


Figure 3-12. Nonpath objects.

Any nonpath objects that are completely surrounded by the path are removed. Based on the path objects found in Figure 3-12, the resulting image is shown in Figure 3-13. Since none of the two nonpath objects were completely surrounded by the path, both were kept.

Figure 3-13. Kept nonpath objects not surrounded by path.

Step 6: Save Largest Horizontal Path Segments

Each row of the image from Step 5 is analyzed, and only the largest connected path segment is saved. The rest of the pixels in the row are classified as nonpath. This step is conducted to remove any smaller paths that are connected or fork out from the main path. The result of this procedure performed on the image in Figure 3-13 is shown in Figure 3-14. This final processed image is used for vehicle navigation.

Figure 3-14. Saved largest horizontal path segments.

Step 7: Sample Window to Determine New Threshold Values

Once the main path is identified in the current image, a sample of the path pixels is statistically analyzed to determine new threshold values for the next frame. A sample window is chosen in the center, lower portion of the image due to a higher probability of most of the window lying on the path. The sample window displayed on the current image is shown in Figure 3-15.

Figure 3-15. Sample window.

Pixels are sampled at specified row and column intervals within the window. If the sample pixel was classified as "path" in Step 6, its red, green, and blue values are used in the statistical analysis. A separate mean and standard deviation is calculated for each red, green, and blue data set from the sampled path pixels. Using a Student t-Distribution (Figliola and Beasley, 2000) for each of the red, green, and blue samples, a confidence interval is selected to get the lower and upper threshold limits for each of the pixel colors.

Sampling path pixels every tenth row and tenth column in the window shown in Figure 3-15 gave a total of $n = 231$ samples and the following mean ($\overline{X}$) and standard deviation (S) values for red, green, and blue:

$$\overline{X}_{Red} = 103.296 \qquad S_{Red} = 12.6491$$

$$\overline{X}_{Green} = 91.3739 \qquad S_{Green} = 11.7047$$

$$\overline{X}_{Blue} = 60.587 \qquad S_{Blue} = 11.7047$$

A confidence interval of 99% was chosen to cover a large range of path pixels. For 99% confidence, a Student t-table gives the t-estimator $t_{.99} = 2.576$. The interval (μ) for this confidence level is given by:

$$\overline{X} - t_{.99}\frac{S}{\sqrt{n}} \le \mu \le \overline{X} + t_{.99}\frac{S}{\sqrt{n}}$$

Plugging in the mean, standard deviation, sample number, and t-estimator, the intervals for each color were found:

$$70 \le \mu_{Red} \le 135$$

$$61 \le \mu_{Green} \le 121$$

$$30 \le \mu_{Blue} \le 90$$

These interval values are saved as the color threshold intervals to use in Step 1 for the next frame.

Path Analysis and Intersection Detection

Once the camera image has been processed, it can be analyzed to find the path center, path edges, and intersections. The path image in Figure 3-16A and its processed threshold image shown in Figure 3-16B will be used to demonstrate the techniques used for path analysis and intersection detection. An external view of the vehicle with a 4-degree angle relative to the path is shown in Figure 3-17. All points and lines shown in the image lie on the ground plane of the vehicle reference frame with the z-coordinate equal to zero (shown by the red plane).

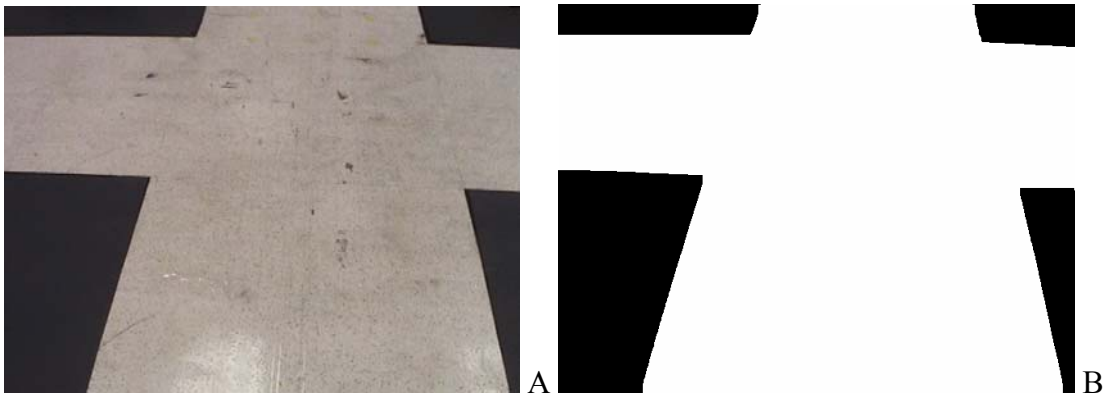


Figure 3-16. Path Image with intersection. A) Raw camera image. B) Processed image.

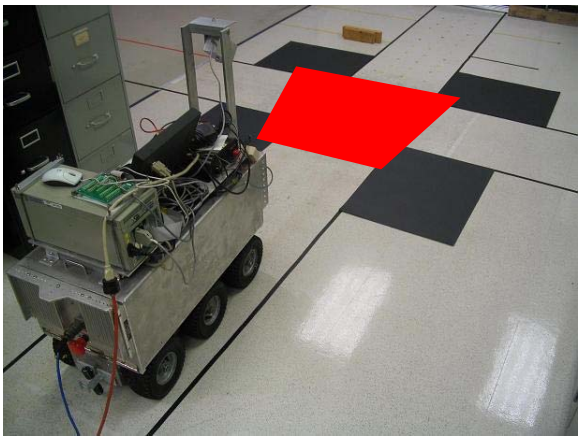


Figure 3-17. Vehicle on path with path view highlighted in red.

Determination of Path Center and Path Edges

A series of equally spaced horizontal lines on the ground in front of the vehicle are projected into the pixel plane to analyze the path. Figure 3-18A shows these path analysis lines on the threshold image from Figure 3-16B. Each line is followed from the left edge of the image to the right. A transition from nonpath (black) to path (white) pixels indicates a left path edge point, and a transition from path to nonpath pixels indicates a right edge point. If both a left and right path edge point are available along these lines, a path center point is calculated, as displayed by the green crosses in Figure 3-18B.

These center points are described in the vehicle coordinate system and are assumed to be on the ground plane with the z-coordinate equal to zero. A best-fit line through these points is found using Least Squares and RANSAC to remove outliers (Fischler and Bolles, 1981). If an intersection was detected in the previous frame, only the center points below the beginning of the intersections (closest to the vehicle) are used to calculate the centerline. An identical procedure with the path edge points is used to estimate the left and right path edges. The estimated centerline and path edges are shown in Figure 3-18C and D.

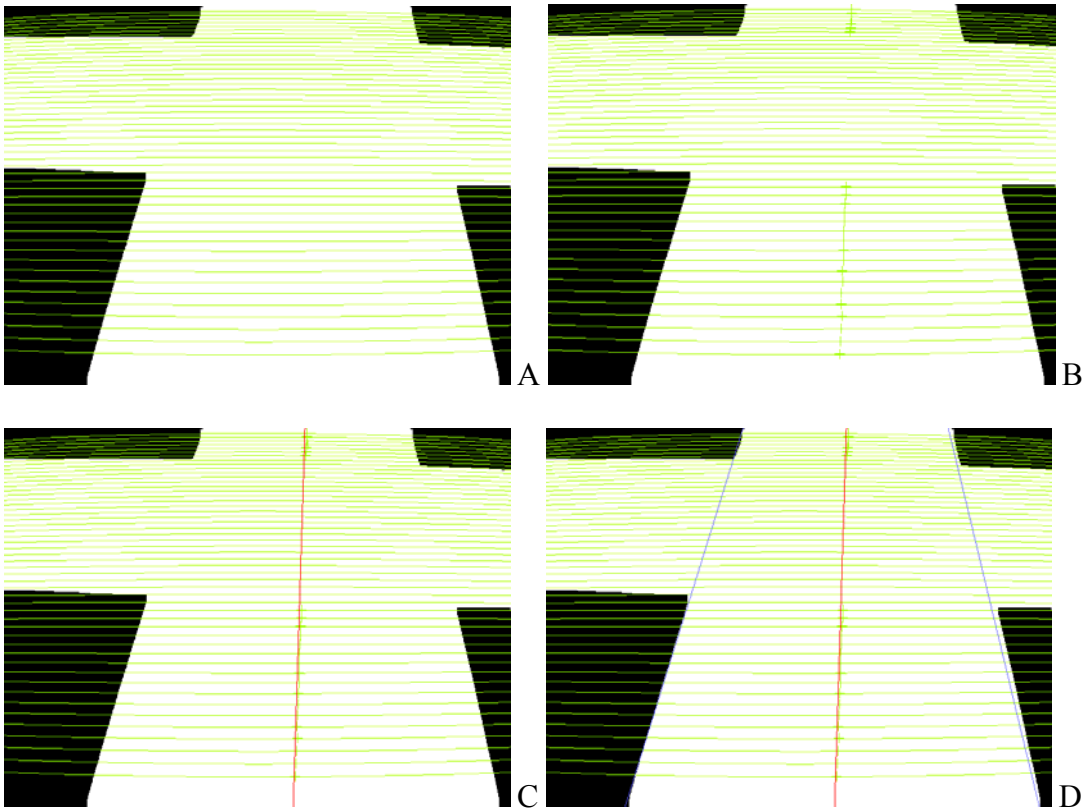


Figure 3-18. Determination of centerline and path edges. A) Path analysis lines shown in green. B) Center points displayed by green crosses. C) Estimated path centerline shown in red. D) Estimated path edges shown in blue.

Intersection Detection

Intersections are detected by finding where the path extends a given distance outside the estimated path edges. This is calculated by creating boundary lines parallel to

the path edge lines and shifted a distance outside the path. Figure 3-19 shows intersection boundary lines (purple lines) shifted 4-inches from the left and right main path edges (blue lines).

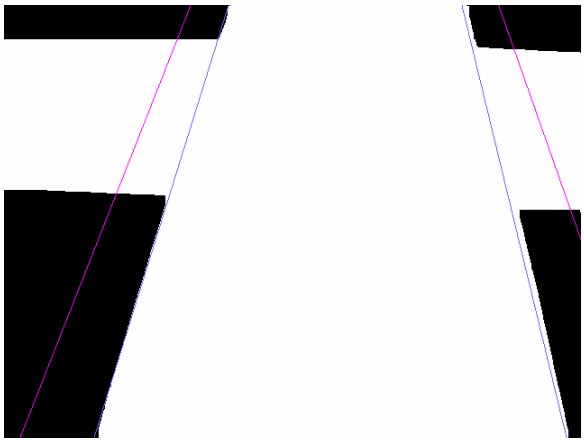


Figure 3-19. Intersection boundary lines shown in purple.

The image is searched along each of these lines starting from the bottom row to the top. Each transition from nonpath (black) to path (white) pixels indicates a possible beginning of an intersection. Each transition from path to nonpath pixels indicates a possible end of an intersection. These points are indicated by the labels "0" and "1" in Figure 3-20.

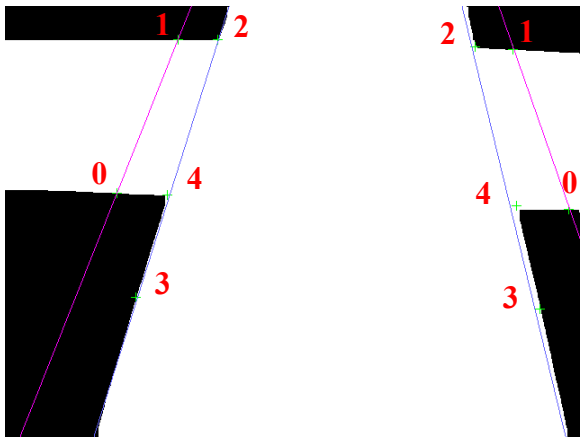


Figure 3-20. Analysis points for intersection detection.

Based on the locations of points "0" and "1," three additional analysis points, "2," "3," and "4", are calculated in Figure 3-20. Point 2 is the intersection of a perpendicular

line drawn from point 1 with the associated path edge. This point is an estimation of the end of the intersection along the main path if the edge of the intersecting path is perpendicular to the main path edge. Using lines 1 (intersection boundary line), 2 (perpendicular intersection path edge), and 3 (main path edge) in Figure 3-21, the location of point 2 is found for the left path edge.

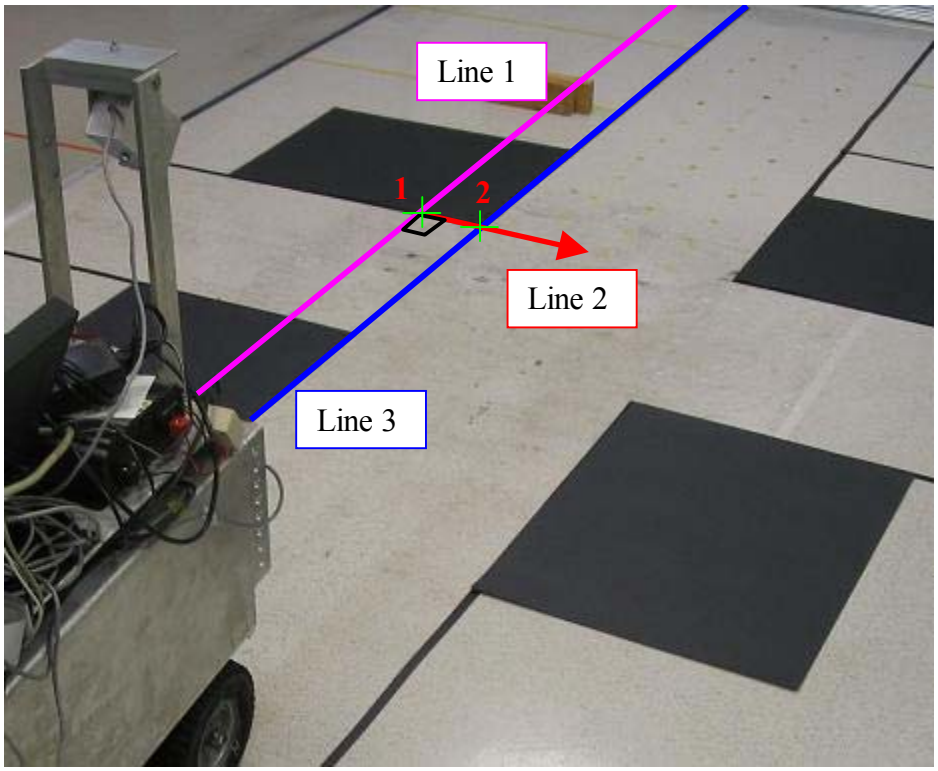


Figure 3-21. Calculation of point 2 during intersection detection.

Given the equation of a line in the vehicle coordinate system

$$A \cdot x + B \cdot y + C \cdot z = D$$

the equations for lines 1 and 2 are

$$A_1 \cdot x + B_1 \cdot y + C_1 \cdot z = D_1$$

$$A_2 \cdot x + B_2 \cdot y + C_2 \cdot z = D_2$$

Assuming both lines lie on the ground plane ($z = 0$), these equations reduce to

$$A_1 \cdot x + B_1 \cdot y = D_1$$

$$A_2 \cdot x + B_2 \cdot y = D_2$$

Line 2 is parallel to line 1 if

$$\frac{-A_2}{B_2} = \frac{B_1}{A_1}$$

Arbitrarily setting $B_2 = 1$, A_2 is found with

$$A_2 = \frac{-B_1}{A_1}$$

Given the coordinate of point 1 as (x_1,y_1,z_1), D_2 is found with

$$D_2 = y_1 - A_2 \cdot x_1$$

The equation of line 3 with the same ground plane assumption is

$$A_3 \cdot x + B_3 \cdot y = D_3$$

Simultaneously solving for x and y in the equations for lines 2 and 3, the coordinates of point 2, (x_2,y_2,z_2), is found from

$$y_2 = \frac{\left(\frac{D_2}{A_2} - \frac{D_3}{A_3}\right)}{\left(\frac{B_2}{A_2} - \frac{B_3}{A_3}\right)}$$

$$x_2 = \frac{D_2}{A_2} - \frac{B_2}{A_2} \cdot y_2$$

$z_2 = 0$ (assuming point 2 is on the ground plane)

Point 4 is the intersection of a perpendicular line drawn from point 0 with the associated path edge. This point is an estimation of the beginning of the intersection along the main path if the edge of the intersecting path is visible by the camera and perpendicular to the main path edge. An identical procedure as that used to find point 2

is conducted to find point 4. The difference is that point 0 is used to solve for the perpendicular line 5, and point 4 is the intersection of line 5 with line 3 (as shown in Figure 3-22).

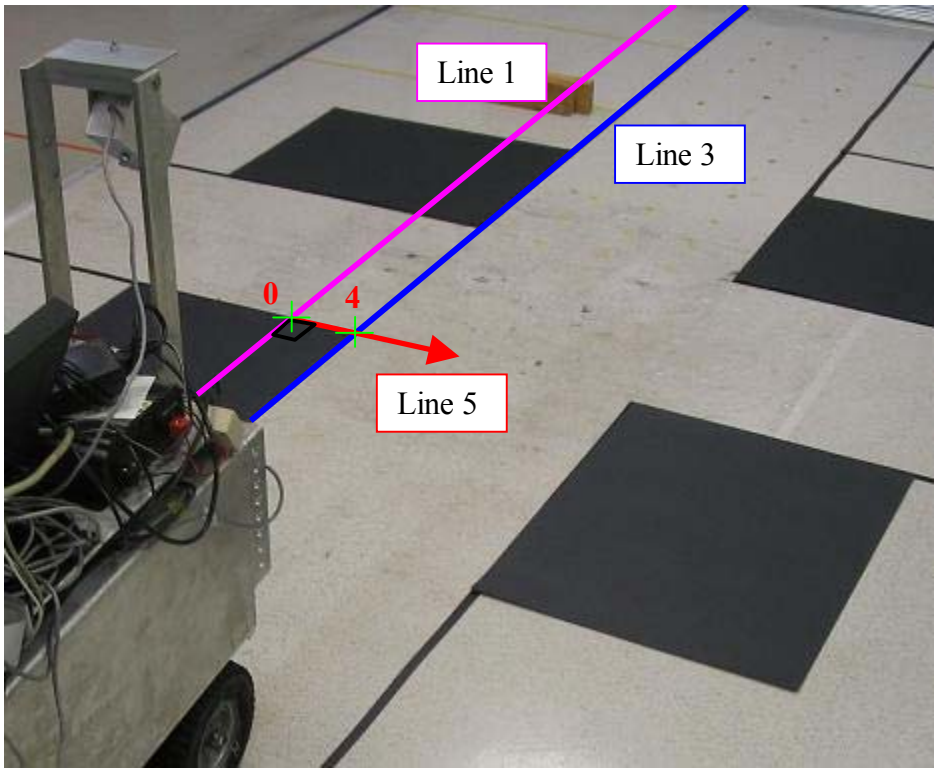


Figure 3-22. Calculation of point 4 during intersection detection.

Point 3 is an estimation of the beginning of the intersection along the main path if the corner of the intersection is a vertical object such as a wall or table leg. If a line is drawn from point 0 to the camera origin, C_0, (represented in yellow as line 6 in Figure 3-23), this line represents all points in the vehicle coordinate system that project into the pixel coordinates of point 0 in the pixel plane. Letting point 0' be the point along this line directly above line 3 (the path edge), point 3 is the intersection of the vertical line drawn from point 0 with line 3.

Point 3 is found mathematically by locating the intersection of the projection of line 6 onto the ground plane (shown as line 4 in Figure 3-23) and finding where this line

intersects the path edge (line 3). The equation for line 4 with the ground plane assumption is

$$A_4 \cdot x + B_4 \cdot y = D_4$$

Given the coordinates of point 0, (x_0,y_0,z_0), and arbitrarily setting $B_4 = 1$, A_4 is found with

$$A_4 = \frac{(y_0 - y_C)}{(x_0 - x_C)}$$

Projecting the camera origin onto the ground plane to get $C_0' = (x_C,y_C,0)$, D_2 is located with

$$D_4 = y_C - A_4 \cdot x_4$$

Simultaneously solving for x and y in the equations for lines 3 and 4, the coordinates of point 3, (x_3,y_3,z_3), are found from

$$y_3 = \frac{\left(\frac{D_4}{A_4} - \frac{D_3}{A_3}\right)}{\left(\frac{B_4}{A_4} - \frac{B_3}{A_3}\right)}$$

$$x_3 = \frac{D_3}{A_3} - \frac{B_3}{A_3} \cdot y_3$$

$z_3 = 0$ (assuming point 3 is on the ground plane)

Figure 3-24 demonstrates how point 3 represents the beginning of an intersection when a vertical object is present at the corner. Boxes were used to simulate vertical corners, as shown in Figure 3-25.

Once analysis points 0 through 4 have been determined, the image is searched for the actual intersection corners. The beginning of the intersection is searched for first within an image window bounded by the columns of points 0 to 4 and the rows of points

1 to 3 (shown by the turquoise rectangles in Figure 3-26). A vertical search for path edges is conducted along the columns in this window, as shown in Figure 3-26A, and a best-fit line is drawn through these points using Least Squares and RANSAC, as shown in Figure 3-26B. Next, a horizontal search for path edges is performed along the rows in this window, as shown in Figure 3-27A, and another best-fit line is drawn through these points using Least Squares and RANSAC, as shown in Figure 3-27B. Whichever of the two lines is closest to point 0 is chosen as the correct line leading to the intersection corner. In this example, the vertical search passes closest to point 0 and is chosen. The location of the beginning of the intersection is the point of intersection between this line and the path edge line (marked by the green crosses in Figure 3-29).

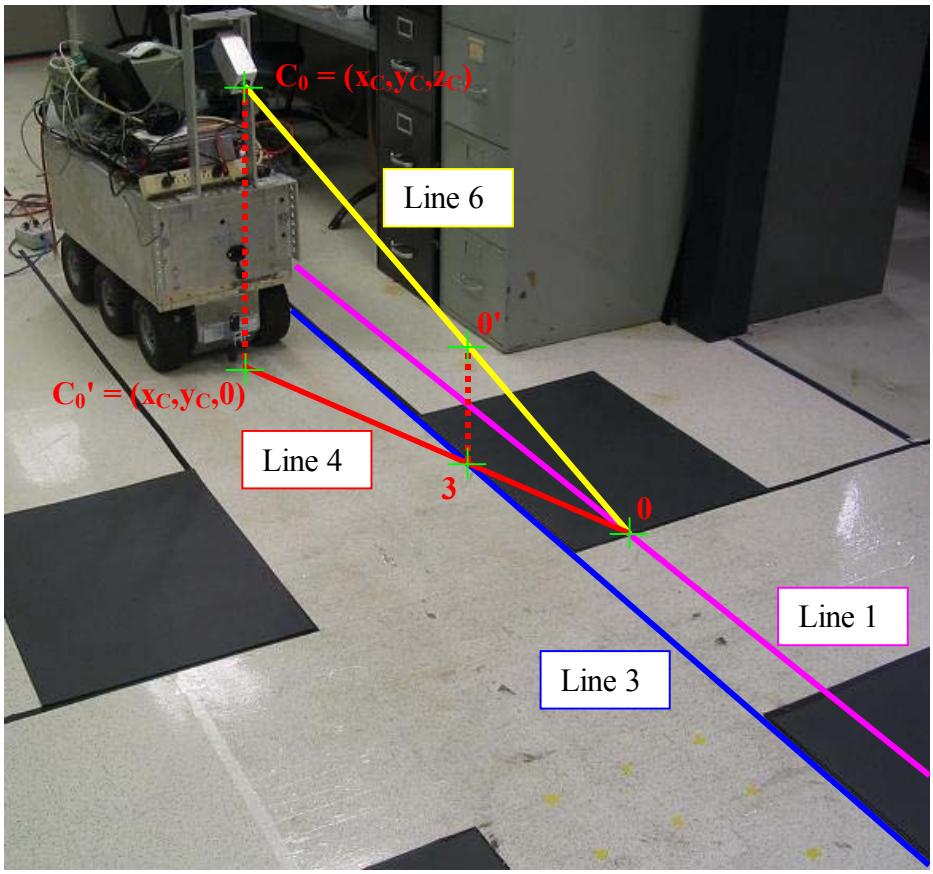


Figure 3-23. Calculation of point 3 during intersection detection.

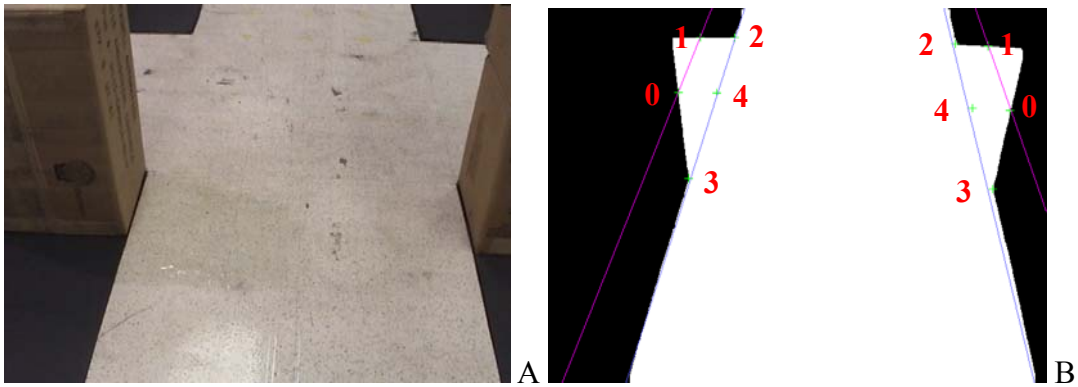


Figure 3-24. Intersection analysis points for vertical corners. A) Original image. B) Analysis points for intersection detection.

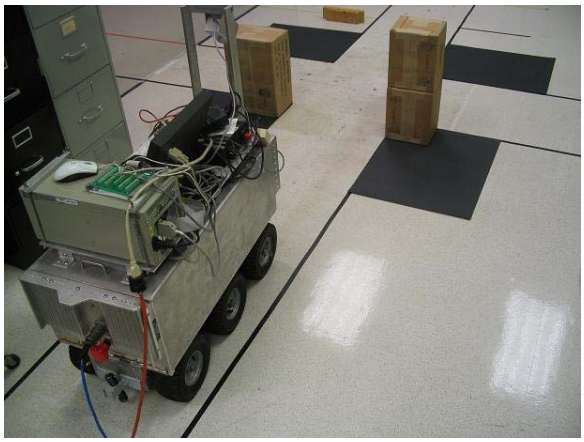


Figure 3-25. Intersection with vertical corners.

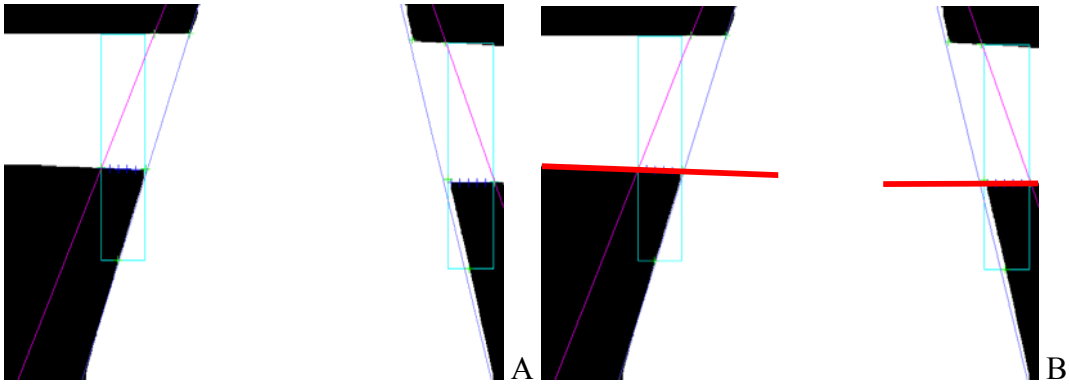


Figure 3-26. Vertical search for beginning of intersection. A) Detected edge points (blue crosses). B) Best-fit lines for left and right path edges (red lines).

The end of the intersection is searched for within an image window bounded by the columns of points 1 to 2 and the rows of points 2 to 4 (shown by the turquoise rectangles in Figure 3-28). Only a vertical search for path edges is performed along the columns in

this window, as shown in Figure 3-28A, and a best-fit line is drawn through these points using Least Squares and RANSAC, as shown in Figure 3-28B. The location of the end of the intersection is the point of intersection between this line and the path edge line (marked by the red crosses in Figure 3-29).

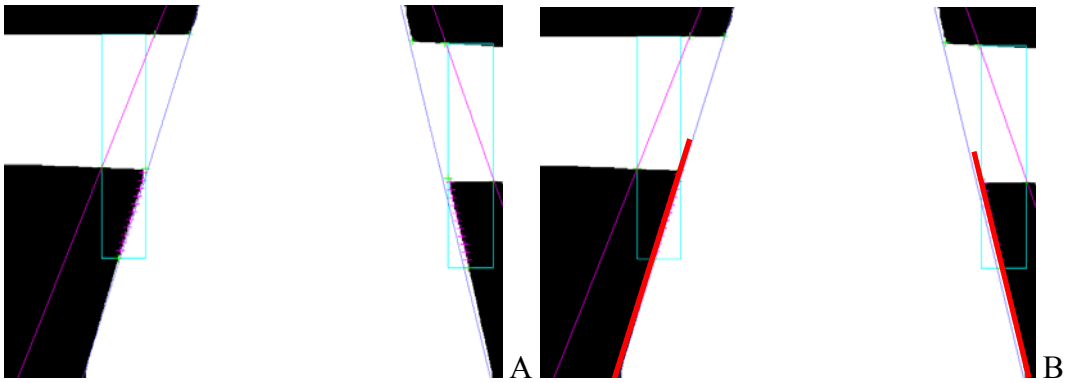


Figure 3-27. Horizontal search for beginning of intersection. A) Detected edge points (purple crosses). B) Best-fit lines for left and right path edges (red lines).

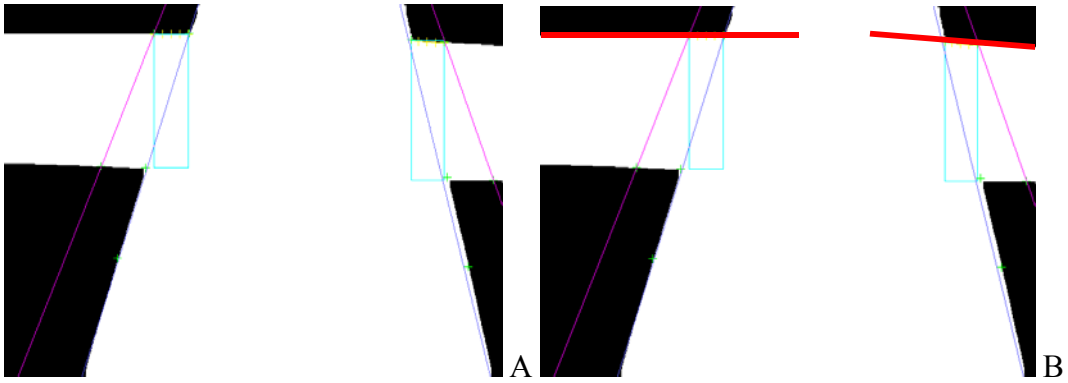


Figure 3-28. Vertical search for end of intersection. A) Detected edge points (blue crosses). B) Best-fit lines for left and right path edges (red lines).

Figure 3-30 demonstrates the intersection detection search results for an intersection with vertical corners at the beginning of the intersection, as well as the final estimated intersection points.

An intersection is valid if the distance between the beginning and end points is greater than a specified value. Figure 3-31 and Figure 3-32 show the final screen shots from path analysis of both intersections with flat corners and vertical corners. For a more

realistic comparison of the calculated intersections points to the actual, the path analysis is displayed on top of the actual image instead of the threshold image.

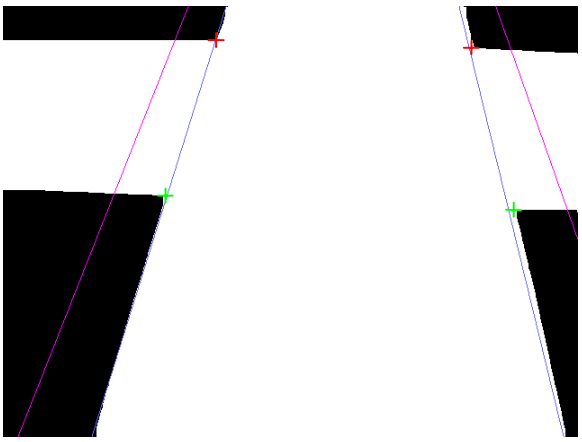


Figure 3-29. Calculated beginning (green crosses) and end (red crosses) of intersection.

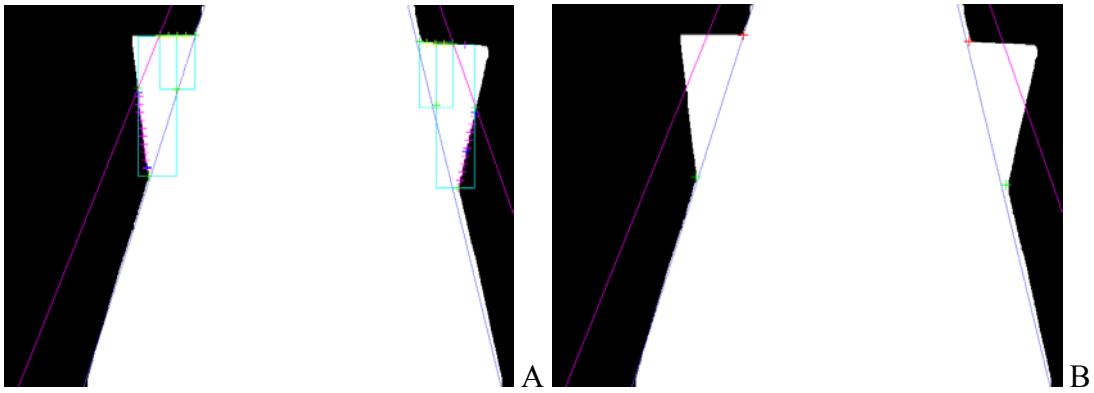


Figure 3-30. Intersection detection for vertical intersection corners. A) Vertical and horizontal search results for intersection beginning and end. B) Calculated beginning (green crosses) and end (red crosses) of intersection.

The coordinates of the beginning and end of the left intersection in Figure 3-31 were determined to be (-8.64, 49.8, 0.00) and (-6.93, 74.2, 0.00) inches in the vehicle coordinate system. The valid intersection width was set as 5 inches. Since the distance between the beginning and end points was calculated as 24.5 inches, this intersection was determined to be valid.

The coordinates of the beginning and end of the left intersection in Figure 3-32 were determined to be (-8.61, 49.1, 0.00) and (-6.71, 74.2, 0.00) inches in the vehicle

coordinate system. Since the distance between the beginning and end points was calculated as 25.2 inches, this intersection was also determined to be valid. Note for both intersection cases, the actual intersection width was 24.0 inches.

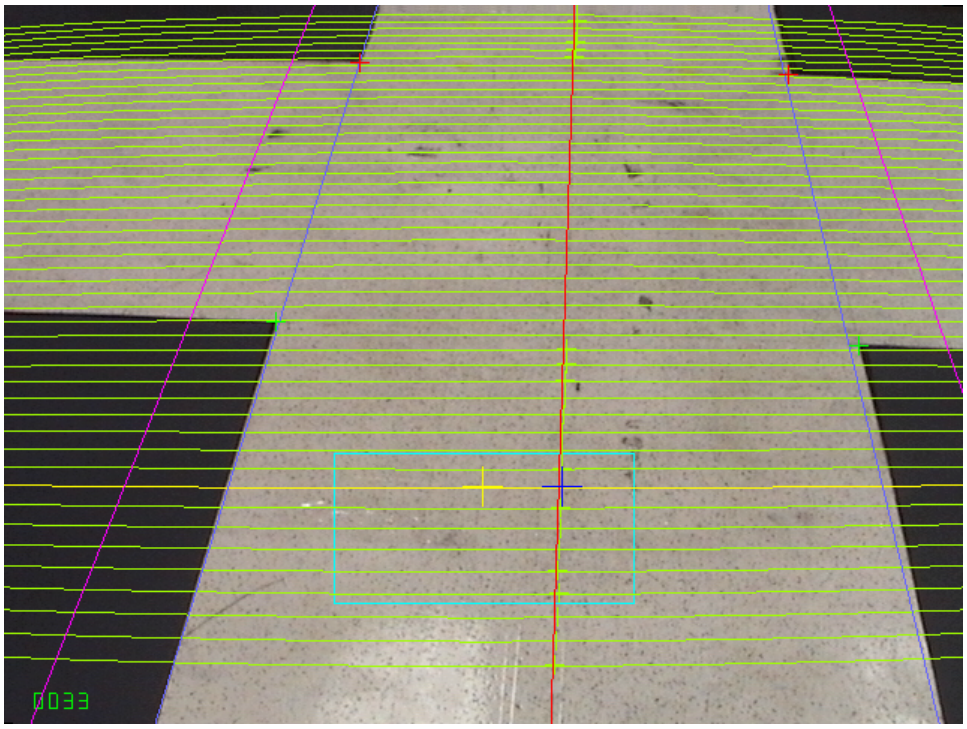


Figure 3-31. Screen view with final path analysis displayed for flat intersection corners. The beginning and end of the intersections are marked by the green and red crosses along the blue path edge lines.

Path Following

The path following algorithm allows various ways to instruct the vehicle to follow the path:

1. Follow the path center
2. Follow an edge
3. Follow the estimated path center when the path center is not visible
4. Follow an estimated edge when the edge is not visible

All control strategies are carried out on a control line a specified lookahead distance in front of the vehicle. The control line is shown in yellow in Figure 3-33. Path

error is defined along this line in inches in the vehicle coordinate frame. The lookahead distance used for the vehicle was set to 40-inches to allow guidance with enough path visible beneath the control line when intersections are detected above the control line.

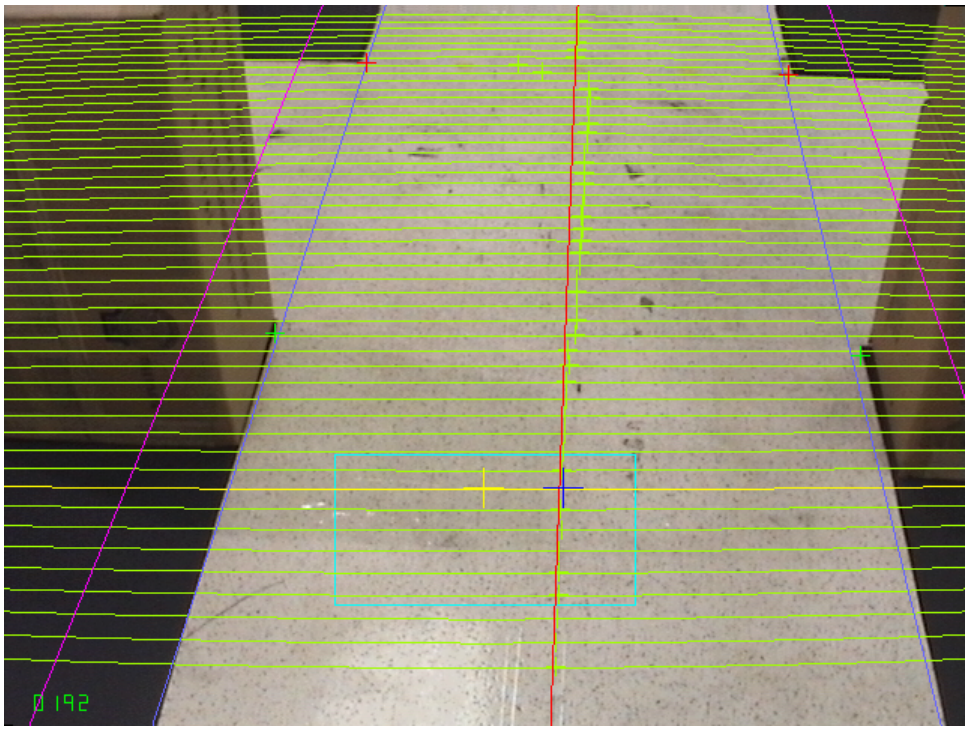


Figure 3-32. Screen view with final path analysis displayed for vertical intersection corners. The beginning and end of the intersections are marked by the green and red crosses along the blue path edge lines.

Following the Path Center

When following the path center, the goal is to reduce the path error between the path center point and desired center point as displayed in Figure 3-33. The path center point is the intersection of the path centerline (drawn in red) with the control line (drawn in yellow). The desired center point is the direct heading of the vehicle along the control line. The path error is the x-coordinate of the desired center point minus the x-coordinate of the path center point in the vehicle coordinate frame.

Follow an Edge

When following an edge, the goal is to reduce the path error between a detected path edge point and desired path edge point. Figure 3-34 demonstrates following of the right path edge. The detected path edge point is the intersection of the path edge (drawn in blue) with the control line (drawn in yellow). The desired path edge point is the location along the control line where the path edge should be for the vehicle to drive a specified distance from that edge. The path error is the x-coordinate of the desired path edge point minus the x-coordinate of the detected path edge point in the vehicle coordinate frame.

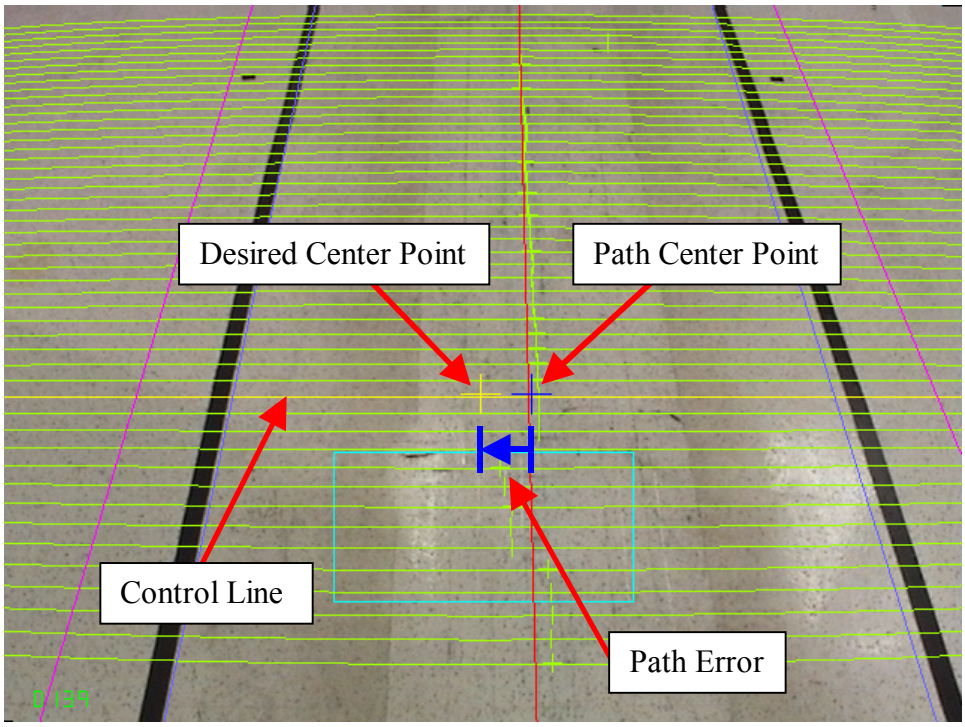


Figure 3-33. Following the path center.

Following the Estimated Path Center

If the path center cannot be found in the current image and the vehicle was instructed to follow the path center, an estimated path centerline is used instead. The estimated path centerline is calculated by shifting any detected path edge half the path

width. Figure 3-35 shows this technique using the estimated path centerline (dark red line) created from shifting the left path edge (blue line) and the assumption that the path is 24-inches wide. An estimate of the path width must be specified in advance. When following the estimated path center, the goal is to reduce the path error between the estimated path center point and desired center point. The estimated path center point is the intersection of the estimated path centerline (drawn in dark red) with the control line (drawn in yellow). The desired center point is the direct heading of the vehicle along the control line. The path error is the x-coordinate of the desired center point minus the x-coordinate of the estimated path center point in the vehicle coordinate frame.

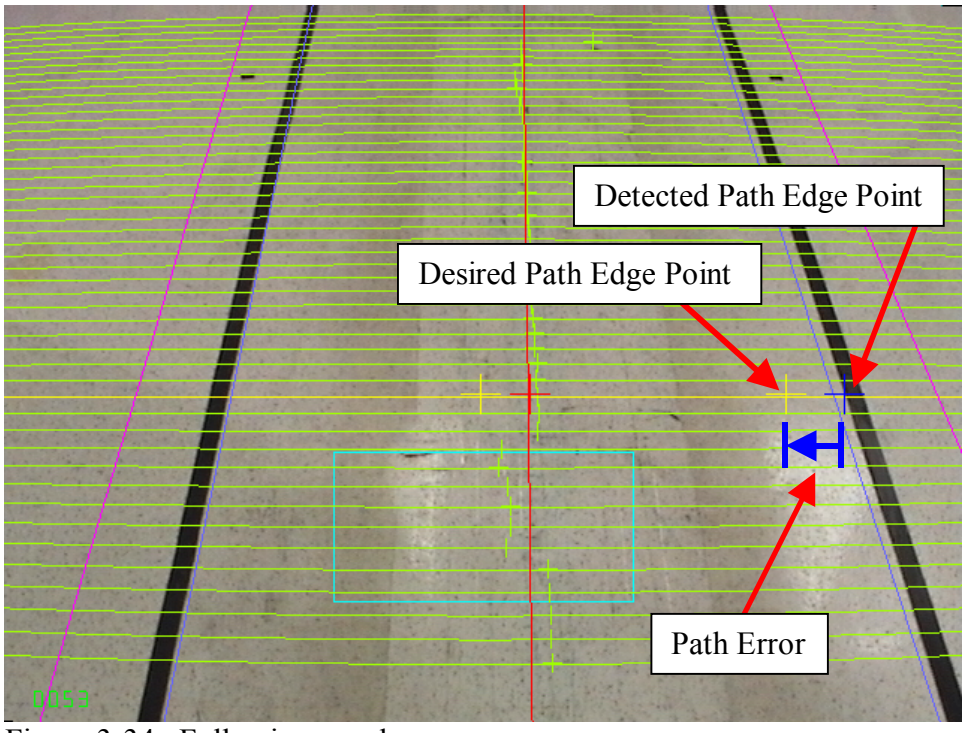


Figure 3-34. Following an edge.

Following an Estimated Path Edge

If the desired path edge to follow cannot be found in the current image, an estimate of that path edge is used instead. The estimated path edge is calculated by shifting the other path edge (if available) the full path width. Figure 3-36 shows this technique using

the estimated right path edge (dark blue line) created from shifting the left path edge (blue line) and the assumption that the path is 24-inches wide. An estimate of the path width must be specified in advance. When following the estimated path edge, the goal is to reduce the path error between the estimated path edge point and desired path edge point. The estimated path edge point is the intersection of the estimated path edge (drawn in dark blue) with the control line (drawn in yellow). The desired path edge point is the location along the control line where the path edge should be for the vehicle to drive a specified distance from that edge. The path error is the x-coordinate of the desired path edge point minus the x-coordinate of the estimated path edge point in the vehicle coordinate frame.

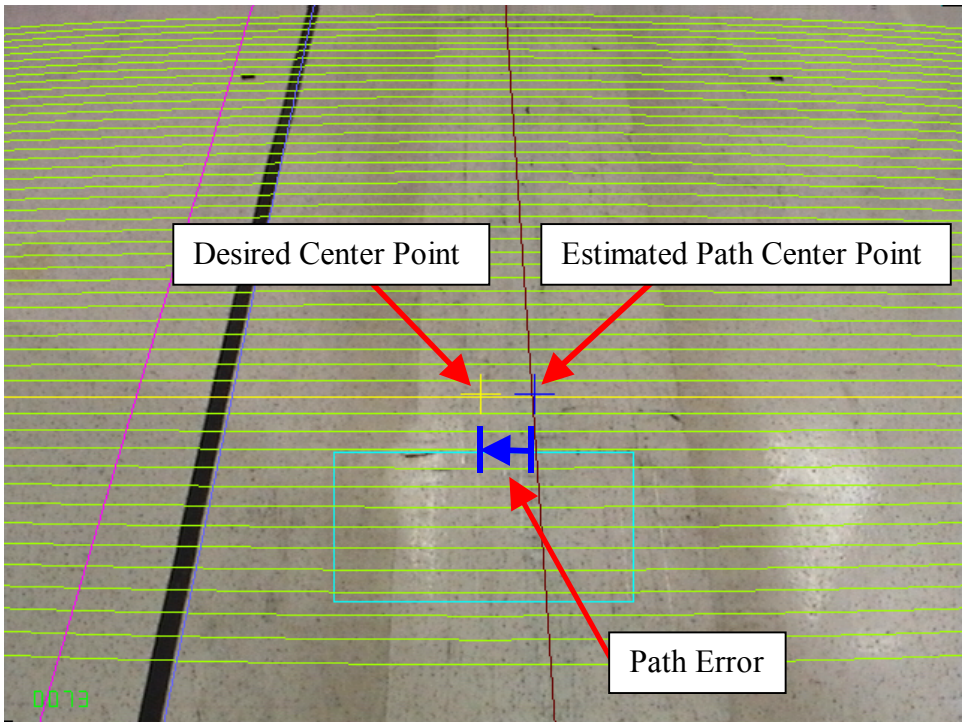


Figure 3-35. Following the estimated path center.

PID Control

A PID controller was developed to direct the vehicle along the path. A block diagram of the vehicle control loop, based on the control architecture pictured in Figure

3-1, is shown in Figure 3-37. The symbol x represents the position of the vehicle along the path, while x_d represents the desired position. The path error, e, is the difference between x and x_d, and becomes the input to the controller. The controller produces a control signal, δ, which is sent to the amplifiers. A block diagram of the controller is shown in Figure 3-38.

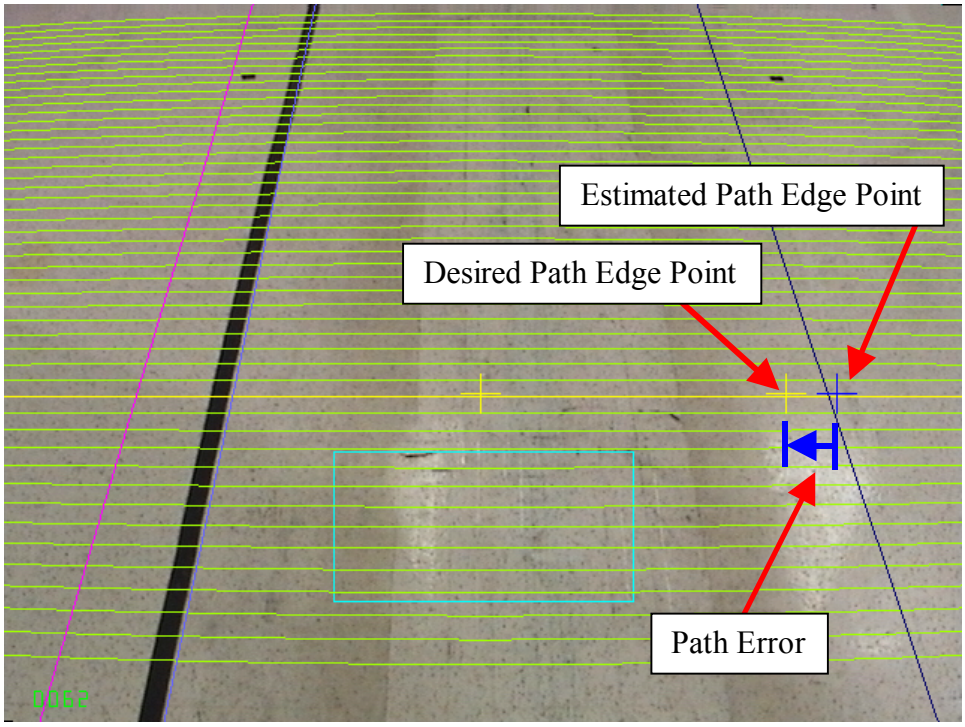


Figure 3-36. Following an estimated path edge.

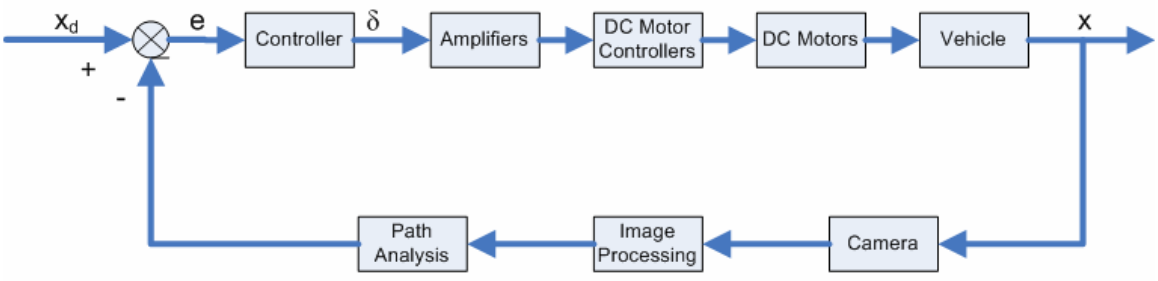


Figure 3-37. Vehicle control loop.

The PID controller consists of three gains: a proportional gain (*Kp*), a derivative gain (*Kd*), and an integral gain (*Ki*). The PidOutput value is calculated from the following equation:

$$PidOutput = Kp \cdot e + Kd \cdot \frac{de}{dt} + Ki \cdot \int edt$$

where *e* is the path error, $\frac{de}{dt}$ is the derivative of the path error, and $\int edt$ is the integral of the path error. $\frac{de}{dt}$ is calculated by finding the difference between the path error of the current frame and the path error of the last frame, and dividing it by the time step between the two frames. $\int edt$ is calculated by multiplying the path error of the current frame with the time step between the current and last frame and continuously summing itself with those of the last frames. $\int edt$ is reset each time the vehicle begins a new path. Through experimentation, the following PID gains were set:

Kp = 10

Kd = 0.625

Ki = 3

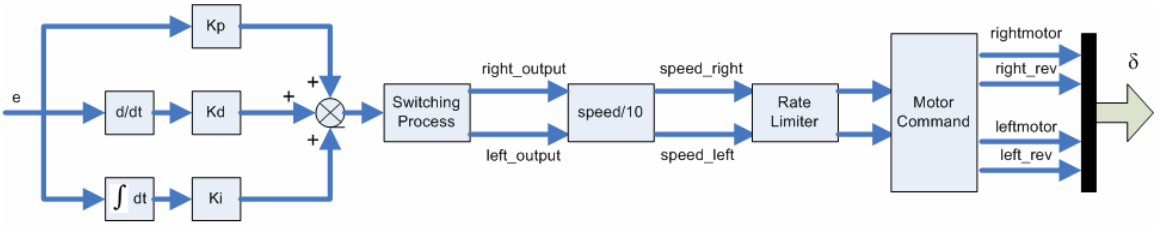


Figure 3-38. Controller block diagram.

A positive PidOutput value instructs the vehicle to turn left, while a negative PidOutput value instructs the vehicle to turn right. Based on the PidOutput, a value between –10 to 10 is given to the left (left_output) and right (right_output) wheel motors.

A negative value turns the wheels backwards, while a positive value turns the wheels forward. To assign these values to the left and right motors, a set of rules are followed. This is called the switching process. When PidOutput is positive, right_output and left_output are assigned from:

```
if (PidOutput >= 20)
        {
        right_output = 5;
        left_output = -1;
        }
if ((PidOutput > 10) & (PidOutput < 20))
        {
        right_output = 10;
        left_output = 0;
        }
if (PidOutput <= 10)
        {
        right_output = PidOutput;
        left_output = 10 – PidOutput;
        }
```

When PidOutput is negative, right_output and left_output are assigned from:

```
if (PidOutput <= -20)
        {
        right_output = -1;
```

```
        left_output = 5;
        }
    if ((PidOutput < -10) & (PidOutput > -20))
        {
        right_output = 0;
        left_output = 10;
        }
    if (PidOutput <= 10)
        {
        right_output = 10 + PidOutput;
        left_output = -PidOutput;
        }
```

When PidOutput = 0, full speed is given to both motors:

```
right_output = 10;
left_output = 10;
```

A desired speed can be specified for the vehicle travel. In this case, the left and right speeds calculated above are scaled:

```
speed_right = right_output*speed/10;
speed_left = left_output*speed/10;
```

To decrease the effects of noise and smooth out the control signal, a ramp limit, speed_ramp_inc, is set on how much the left and right speeds, speed_right and speed_left, can increase or decrease. To accomplish this, the current speed values,

speed_right and speed_left, are compared to the last speed values, speed_right_last and speed_left_last. The right speed is adjusted with the following code:

```
if (speed_right > (speed_right_last + speed_ramp_inc))
        {
        speed_right = speed_right_last + speed_ramp_inc;
        }
 if (speed_right < (speed_right_last - speed_ramp_inc))
        {
        speed_right = speed_right_last - speed_ramp_inc;
        }
```

The left speed is accomplished in the same fashion. A ramp limit of 0.5 was used for experimentation. The values of speed_right and speed_left range from –10 to 10.

The final control signal output from the computer to the amplifiers consist of voltages for the left and right motors, leftmotor and rightmotor, ranging from 0 to 2.5 Volts and directions for each motor, left_rev and right_rev. These are outputted from the controller in Figure 3-37and Figure 3-38 as δ. For left_rev and right_rev, a value of 1 commands the motor to drive forward, and a value of 0 commands the motor to drive backwards. The following code accomplishes this for the left motor

```
if (speed_left >= 0)
        {
        left_rev = 1;
        leftmotor = speed_left/4.0;     // Range: 0 to 2.5V
        }
```

```
else
        {
        left_rev = 0;
        leftmotor = -speed_left/4.0;    // Range: 0 to 2.5V
        }
```

Visual Odometer

A visual odometer was developed to estimate vehicle position and orientation relative to the world coordinate system. The odometer determines vehicle movement by tracking features on the ground. The following steps are carried out in this process:

1. Initialize odometer
2. Initial search for features
3. Tracking features
4. Determination of position and orientation change

Initialize Odometer

When the odometer is initialized, the world coordinate system is defined relative to the vehicle coordinate system. All translations and rotations in subsequent images will be relative to this world system.

Initial Search for Features

The odometer relies on tracking features through a set of image frames. When starting the odometer, an initial set of features must be found for tracking. The yellow box in the upper-center area of the image in Figure 3-39A shows where initial features are found. The KLT search box is 151 pixels wide, 31 pixels high, and located at pixel row 380 and column 320. These specifications were chosen to allow pixels to be tracked a large distance when the vehicle is moving forward. The KLT search box size was

specified because searching the entire image for features would take too much computation time.

Within the search box, good features are defined as corners found using the KLT algorithm (Csetverikov, 2004). 7x7 pixel sizes were chosen as feature sizes, because they are small enough for fast searching and large enough to uniquely define the feature. The strongest corner features found by the KLT algorithm are shown as red boxes within the search box in Figure 3-39A. Because tracking features takes significant computation time, the number used in the tracking process is limited to a minimum. A minimum of two features is needed to define the vehicle motion relative to the ground. The five strongest features found with the KLT algorithm, represented by the green boxes in Figure 3-39B, are used for tracking.

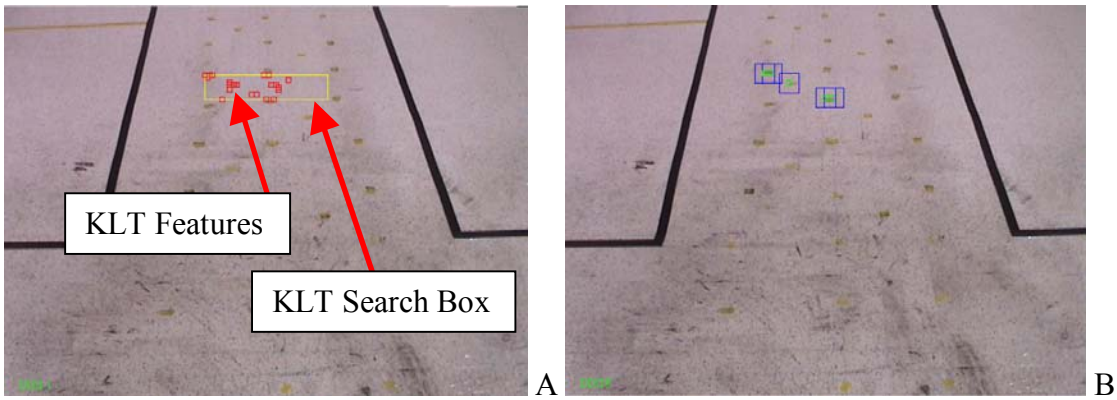


Figure 3-39. Initial search for features. A) Good features found in search box. B) Five strongest KLT features selected for tracking.

Tracking Features

Figure 3-40 demonstrates feature tracking in two subsequent images. The small green boxes mark the five 7x7 pixel features being tracked. Features are tracked by saving the 7x7 pixel feature in the first image and looking for a 7x7 pixel area that best matches it in the second image. Only the area within the 25x25 pixel blue box is searched, because the feature is assumed not to move very far from one image to another.

The location of the feature search box is based on the movement of that feature found in the last frame.

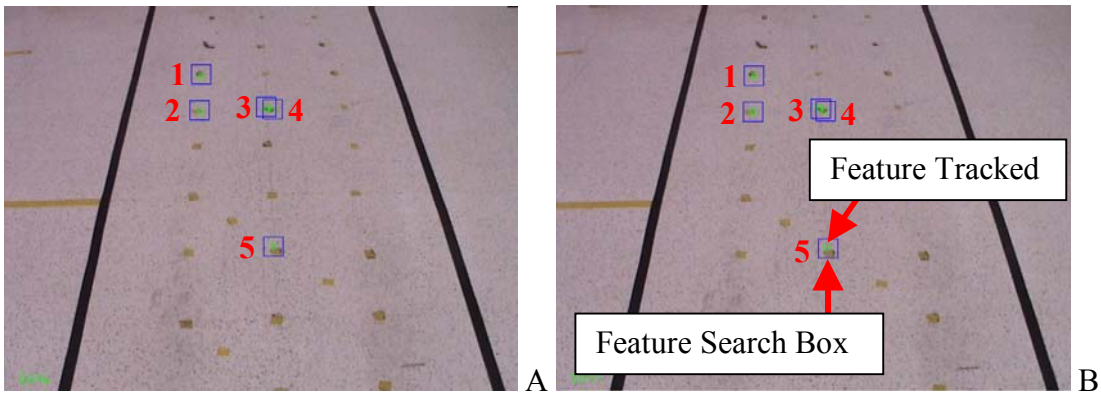


Figure 3-40. Tracking features. A) First frame. B) Second frame.

To locate the features defined in image I_1 (Figure 3-40A) in image I_2 (Figure 3-40B), the sum-of-squared difference (SSD) is calculated for each possible 7x7 pixel area in the search box using the following equation (Barron et al., 1993):

$$SSD_{1,2}(x;d) = \sum_{j=-n}^{n} \sum_{i=-n}^{n} W(i,j)\left[I_1\left(x+(i,j)\right) - I_2\left(x+d+(i,j)\right)\right]^2$$

where W is a discrete 2-d window function, x is the center pixel coordinates of the feature defined in frame 1, n = *floor(feature pixel width/2)*, and d is the pixel row and column shift used to move the 7x7 pixel area within the search box in image I_2. The best match gives the least sum-of-squared difference. Figure 3-41 shows a larger view of the five features tracked from the first and second frame in Figure 3-40.

The restraint made on the features for the visual odometer is that they must be fixed on the ground plane. Conditions that must be avoided when tracking features are:

1. The feature tracked is not on the ground plane
2. The feature moves independent of the ground plane
3. The feature is lost or not correctly tracked

Figure 3-41. Large view of features matched in Figure 3-40. A) Features 1 through 5 in Figure 3-40A. B) Features 1 through 5 in Figure 3-40B. Note the outer feature pixels are not visible due to the green box drawn around the feature.

To prevent these conditions from occurring during feature tracking, a geometric relationship amongst the features is defined from frame to frame. The distance from each feature to all other features is measured and recorded. Figure 3-42 shows the distance calculated between features 1 and 5. ${}^{1}F_{1}$ and ${}^{1}F_{5}$ represent the coordinates of features 1 and 5 in the vehicle coordinate system in image frame 1. Note that these coordinates assume the features exist on the ground plane as defined in the camera model. ${}^{1}d_{1,5}$ represents the distance between features 1 and 5 in image frame 1. Table 3-1 shows the complete distance measurements between features from frame 1 in Figure 3-39A.

When the features from the first frame are found in the second frame, the distance relationship calculations are repeated. Table 3-3 shows the complete distance measurements between features from frame 2 in Figure 3-39B. Next, the difference between the distance measurements from frame 1 to frame 2 are calculated and shown in Table 3-4.

If all features were fixed to the ground and tracked accurately, the distance relationships between each of the features should remain the same. Analyzing the change in distances between the two frames compared in Table 3-4 show little change, proving all five of the tracked features are valid.

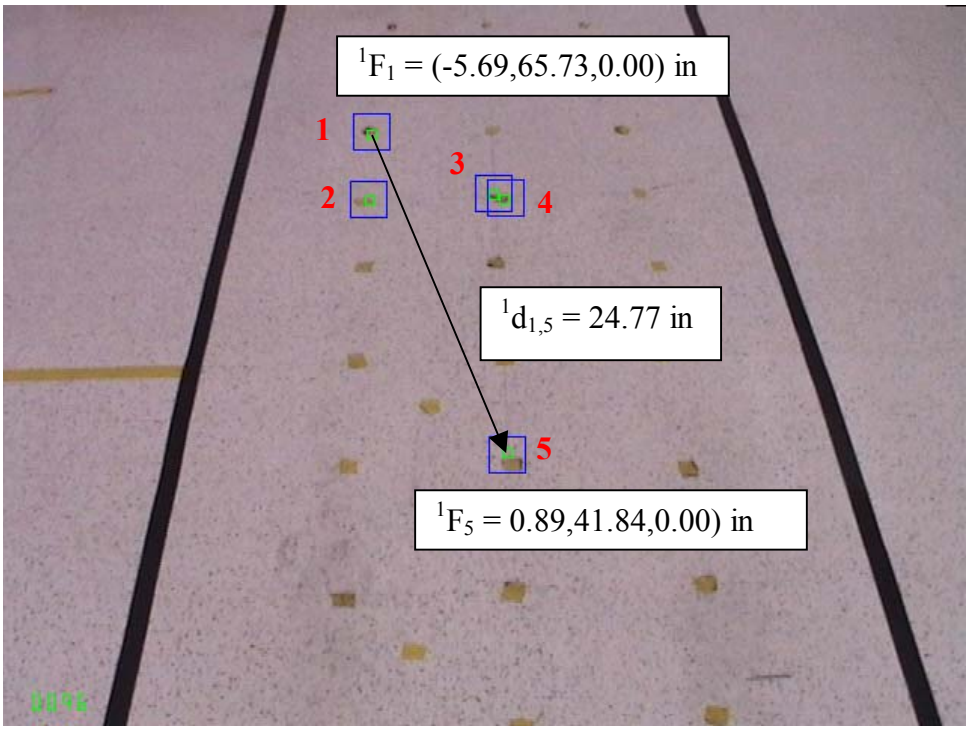


Figure 3-42. Distance between features 1 and 5.

Table 3-2. Distances between features in frame 1 shown in Figure 3-40A in inches.

Feature	**1**	**2**	**3**	**4**	**5**
1	$^1d_{1,1} = 0.00$	$^1d_{1,2} = 6.44$	$^1d_{1,3} = 8.59$	$^1d_{1,4} = 9.23$	$^1d_{1,5} = 24.77$
2	$^1d_{2,1} = 6.44$	$^1d_{2,2} = 0.00$	$^1d_{2,3} = 5.87$	$^1d_{2,4} = 6.35$	$^1d_{2,5} = 18.52$
3	$^1d_{3,1} = 8.59$	$^1d_{3,2} = 5.87$	$^1d_{3,3} = 0.00$	$^1d_{3,4} = 0.64$	$^1d_{3,5} = 17.96$
4	$^1d_{4,1} = 9.23$	$^1d_{4,2} = 6.35$	$^1d_{4,3} = 0.64$	$^1d_{4,4} = 0.00$	$^1d_{4,5} = 17.55$
5	$^1d_{5,1} = 24.77$	$^1d_{5,2} = 18.52$	$^1d_{5,3} = 17.96$	$^1d_{5,4} = 17.55$	$^1d_{5,5} = 0.00$

Table 3-3. Distances between features in frame 2 shown in Figure 3-40B in inches.

Feature	**1**	**2**	**3**	**4**	**5**
1	$^2d_{1,1} = 0.00$	$^2d_{1,2} = 6.53$	$^2d_{1,3} = 8.54$	$^2d_{1,4} = 9.27$	$^2d_{1,5} = 24.77$
2	$^2d_{2,1} = 6.53$	$^2d_{2,2} = 0.00$	$^2d_{2,3} = 5.85$	$^2d_{2,4} = 6.31$	$^2d_{2,5} = 18.44$
3	$^2d_{3,1} = 8.54$	$^2d_{3,2} = 5.85$	$^2d_{3,3} = 0.00$	$^2d_{3,4} = 0.73$	$^2d_{3,5} = 18.01$
4	$^2d_{4,1} = 9.27$	$^2d_{4,2} = 6.31$	$^2d_{4,3} = 0.73$	$^2d_{4,4} = 0.00$	$^2d_{4,5} = 17.47$
5	$^2d_{5,1} = 24.77$	$^2d_{5,2} = 18.44$	$^2d_{5,3} = 18.01$	$^2d_{5,4} = 17.47$	$^2d_{5,5} = 0.00$

Figure 3-43 demonstrates a case when a feature was not tracked correctly between two consecutive frames. A close-up of the five features being tracked is shown in Figure 3-43C and Figure 3-43D. Figure 3-43B shows how all features were successfully tracked in frame 2 except for feature 1 (marked by the purple square). Feature 1 was determined to be non-valid by performing the distance relationship analysis between features in each

of the two frames. Table 3-5 and Table 3-6 show the complete distance measurements between features from frame 1 in Figure 3-43A and frame 2 in Figure 3-43B. The difference between the distance measurements from frame 1 to frame 2 are calculated and shown in Table 3-7.

Table 3-4. Difference in feature distance in frame 1 (Table 3-1) and frame 2 (Table 3-3) in inches.

Feature	**1**	**2**	**3**	**4**	**5**
1	0.00	0.09	0.05	0.04	0.00
2	0.09	0.00	0.02	0.04	0.08
3	0.05	0.02	0.00	0.09	0.05
4	0.04	0.04	0.09	0.00	0.08
5	0.00	0.08	0.05	0.08	0.00

As seen in Table 3-7, many of the changes of distances measured from feature 1 (the row) changed significantly more than those for the other four features. A threshold value of 0.1 was set to pick out change in feature distances that indicated significant change. Any change of distance ≥ 0.1 is labeled as a significant change. These are marked in bold in Table 3-7. If the number of high changes is $\geq (n/2)$, where n is the number of features, then the feature is labeled as invalid. For the case in Table 3-7, $n/2 = 5/2 = 2.5$. The number of significant changes for feature 1 is 3. Since $3 > 2.5$, the feature is classified as invalid. When a feature is classified as invalid, it is removed from the tracking list and not used in the visual odometer calculations, and a new search is run on the current image to find a new feature to replace it (seen by the new yellow KLT search box in Figure 3-43B).

All features whose search area has moved outside the image are also removed from the tracking list. A new feature search is calculated to find new features to replace them.

Figure 3-44 demonstrates a three-frame sequence in which a feature is lost, replaced, and tracked.

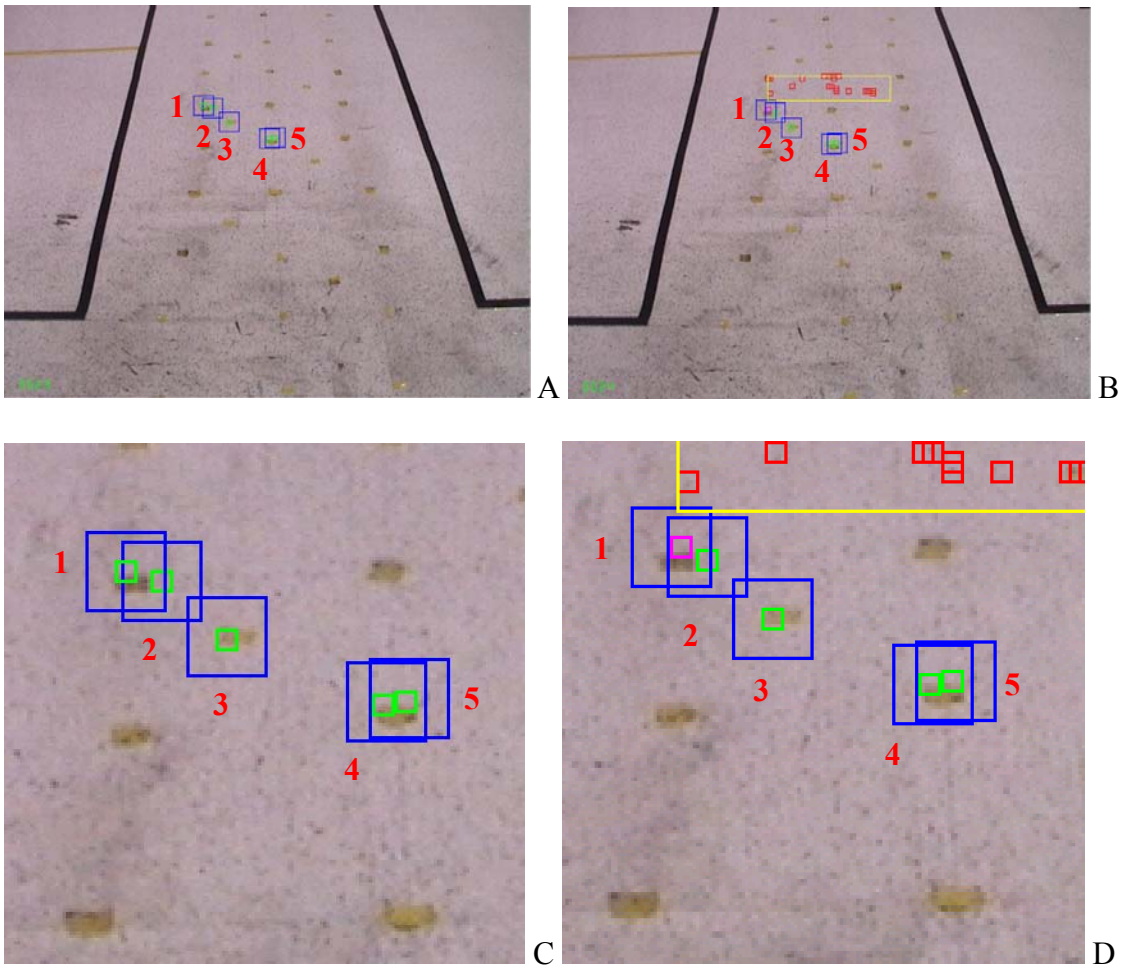


Figure 3-43. Mistracked feature. A) First frame. B) Second frame. C) Close-up of features from first frame. D) Close-up of features from second frame.

Table 3-5. Distances between features in frame 1 shown in Figure 3-43A in inches.

Feature	**1**	**2**	**3**	**4**	**5**
1	$^{1}d_{1,1} = 0.00$	$^{1}d_{1,2} = 0.92$	$^{1}d_{1,3} = 3.67$	$^{1}d_{1,4} = 7.76$	$^{1}d_{1,5} = 8.04$
2	$^{1}d_{2,1} = 0.92$	$^{1}d_{2,2} = 0.00$	$^{1}d_{2,3} = 2.84$	$^{1}d_{2,4} = 6.88$	$^{1}d_{2,5} = 7.14$
3	$^{1}d_{3,1} = 3.67$	$^{1}d_{3,2} = 2.84$	$^{1}d_{3,3} = 0.00$	$^{1}d_{3,4} = 4.15$	$^{1}d_{3,5} = 4.47$
4	$^{1}d_{4,1} = 7.76$	$^{1}d_{4,2} = 6.88$	$^{1}d_{4,3} = 4.15$	$^{1}d_{4,4} = 0.00$	$^{1}d_{4,5} = 0.49$
5	$^{1}d_{5,1} = 8.04$	$^{1}d_{5,2} = 7.14$	$^{1}d_{5,3} = 4.47$	$^{1}d_{5,4} = 0.49$	$^{1}d_{5,5} = 0.00$

Determination of Position and Orientation Change

Based on the movement of the tracked features, the change in vehicle position and orientation is calculated between frames. Given two features tracked successfully from

frame 1 to frame 2, the translation of the vehicle and the rotation of the vehicle about the world z-axis are found. The first calculation performed is the rotation of the features in frame 2 relative to the features in frame 1. Since the features are fixed in the world coordinate system, this is equivalent to finding the rotation for the world system in frame 2 relative to the world system in frame 1. Figure 3-45 shows the calculations of two consecutive frames using the coordinates of two features. ${}^{1}F_1$ and ${}^{1}F_2$ represents the coordinates of features 1 and 2 from frame 1 in the vehicle coordinate system. ${}^{2}F_1$ and ${}^{2}F_2$ represents the coordinates of features 1 and 2 from frame 2 in the vehicle coordinate system. ${}^{V}Y'$ represents a line drawn through feature 1 parallel to the y-axis of the vehicle coordinate system. γ_1 and γ_2 represent the angles between ${}^{V}Y'$ and a ray drawn from feature 1 to feature 2 in the two frames. The rotation of the world system in frame 2 relative to the world system in frame 1, γ, is calculated:

$$\gamma = \gamma_2 - \gamma_1$$

Table 3-6. Distances between features in frame 2 shown in Figure 3-43B in inches.

Feature	**1**	**2**	**3**	**4**	**5**
1	${}^{2}d_{1,1} = 0.00$	${}^{2}d_{1,2} = 0.82$	${}^{2}d_{1,3} = 3.61$	${}^{2}d_{1,4} = 7.65$	${}^{2}d_{1,5} = 7.91$
2	${}^{2}d_{2,1} = 0.82$	${}^{2}d_{2,2} = 0.00$	${}^{2}d_{2,3} = 2.81$	${}^{2}d_{2,4} = 6.82$	${}^{2}d_{2,5} = 7.08$
3	${}^{2}d_{3,1} = 3.61$	${}^{2}d_{3,2} = 2.81$	${}^{2}d_{3,3} = 0.00$	${}^{2}d_{3,4} = 4.12$	${}^{2}d_{3,5} = 4.44$
4	${}^{2}d_{4,1} = 7.65$	${}^{2}d_{4,2} = 6.82$	${}^{2}d_{4,3} = 4.12$	${}^{2}d_{4,4} = 0.00$	${}^{2}d_{4,5} = 0.48$
5	${}^{2}d_{5,1} = 7.91$	${}^{2}d_{5,2} = 7.08$	${}^{2}d_{5,3} = 4.44$	${}^{2}d_{5,4} = 0.48$	${}^{2}d_{5,5} = 0.00$

Table 3-7. Difference in feature distance in frame 1 (Table 3-5) and frame 2 (Table 3-6) in inches.

Feature	**1**	**2**	**3**	**4**	**5**
1	0.00	**0.1**	0.06	**0.11**	**0.13**
2	**0.1**	0.00	0.03	0.06	0.06
3	0.06	0.03	0.00	0.03	0.03
4	**0.11**	0.06	0.03	0.00	0.01
5	**0.13**	0.06	0.03	0.01	0.00

Given the vehicle system coordinates of feature 1 in frame 1, ${}^{1}F_1 = ({}^{1}F_{1,x}, {}^{1}F_{1,y}, {}^{1}F_{1,z})$, and feature 1 in frame 2, ${}^{2}F_1 = ({}^{2}F_{1,x}, {}^{2}F_{1,y}, {}^{2}F_{1,z})$, the translation of the world

system in frame 2 relative to the world system frame 1 is calculated next. Let ${}^{1}T_{2}$ represent the transformation matrix relating the world system in frame 2 to the world system frame 1. The relationship between the coordinates of feature 1 in frame 1 and frame 2 is given by:

$${}^{1}F_{1} = {}^{1}T_{2} \cdot {}^{2}F_{1}$$

Expanding this matrix and assuming vehicle rotation only occurs about its z-axis,

$$\begin{bmatrix} {}^{1}F_{1,x} \\ {}^{1}F_{1,y} \\ {}^{1}F_{1,z} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 & T_x \\ \sin(\gamma) & \cos(\gamma) & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} {}^{2}F_{1,x} \\ {}^{2}F_{1,y} \\ {}^{2}F_{1,z} \\ 1 \end{bmatrix}$$

With the assumption that all points are on the ground ($^{1}F_{1,z}$, $^{2}F_{1,z} = 0$), the translation of the world frame in the vehicle coordinate system in the x, y, and z directions, T_x, T_y and T_z, are solved:

$$T_x = {}^{1}F_{1,x} - \left(\cos(\gamma) \cdot {}^{2}F_{1,x} - \sin(\gamma) \cdot {}^{2}F_{1,y}\right)$$

$$T_y = {}^{1}F_{1,y} - \left(\sin(\gamma) \cdot {}^{2}F_{1,x} + \cos(\gamma) \cdot {}^{2}F_{1,y}\right)$$

$$T_z = 0$$

To find the best estimates for the rotation, γ, and translation, T_x, T_y, and T_z, between frames, these values are calculated for all combinations of the valid tracked features. For each set of rotation and translation values calculated, ${}^{1}T_{2}$ is composed and multiplied with the coordinates of each of the features, ${}^{2}F_{i}$, to calculate ${}^{2}F'_{i}$.

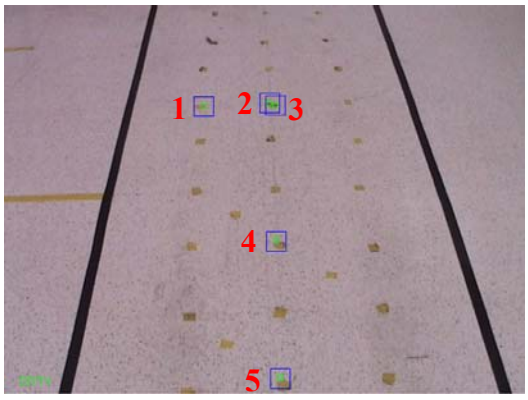
$${}^{1}F_{i}' = {}^{1}T_{2} \cdot {}^{2}F_{i}$$

An error for the estimated set of rotation and translation values is

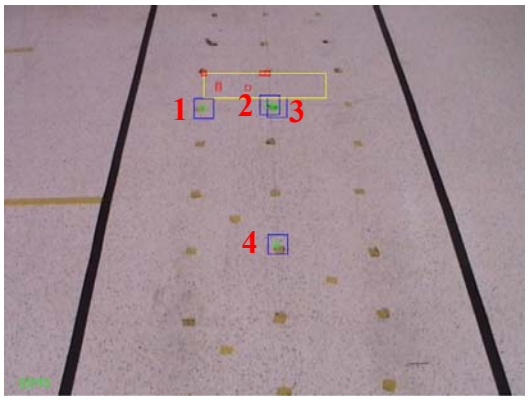
$$error = \sum_{i=1}^{n} ({}^{1}F_i' - {}^{1}F_i)^2$$

where n is the number of valid tracked features available. The set of rotation and translation values yielding the least error is used as the best estimate for γ, T_x, T_y and T_z. As demonstrated before, this set of values is used to develop ${}^{1}T_2$, which represents the transformation matrix relating the world system in frame 2 to the world system frame 1:

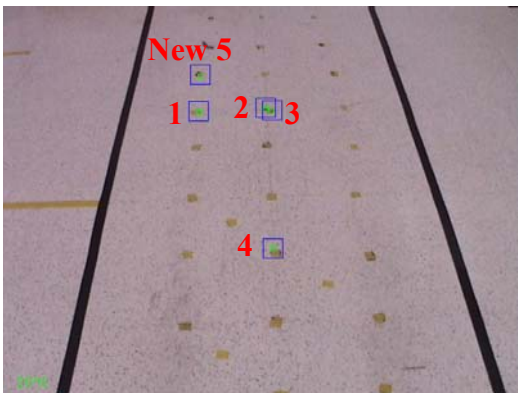
$${}^{1}T_2 = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 & T_x \\ \sin(\gamma) & \cos(\gamma) & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



A



B



C

Figure 3-44. Feature leaving image and replaced. A) Frame 1: Five features tracked. B) Frame 2: feature 5 moved off image and replacement feature search. C) Fame 3: New feature 5 tracked.

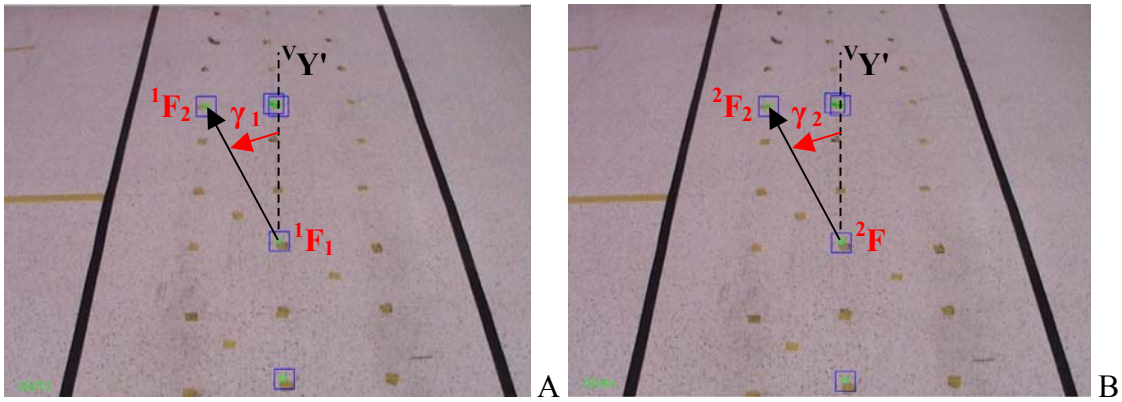


Figure 3-45. Calculation of rotation between two consecutive frames. A) Frame 1. B) Frame 2.

If ${}^{W}T_{V}$ represents the transformation matrix relating the vehicle coordinate system to the world system, a new ${}^{W}T_{V}$ is calculated using the latest ${}^{1}T_{2}$:

$${}^{W}T_{V} = {}^{W}T_{V} \cdot {}^{1}T_{2}$$

The final location of the vehicle in the world coordinate system is ${}^{W}P_{V} = ({}^{W}x_{V}, {}^{W}y_{V}, {}^{W}z_{V})$. The final orientation of the vehicle about the world z-axis is ${}^{W}\gamma_{V}$. Both are extracted from ${}^{W}T_{V}$:

$${}^{W}T_{V} = \begin{bmatrix} \cos({}^{W}\gamma_{V}) & -\sin({}^{W}\gamma_{V}) & 0 & {}^{W}x_{V} \\ \sin({}^{W}\gamma_{V}) & \cos({}^{W}\gamma_{V}) & 0 & {}^{W}y_{V} \\ 0 & 0 & 1 & {}^{W}z_{V} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^{W}\gamma_{V} = a\tan 2(\sin({}^{W}\gamma_{V}), \cos({}^{W}\gamma_{V}))$$

${}^{V}T_{W}$, representing the transformation matrix relating the world system to the vehicle coordinate system, can also be calculated from ${}^{W}T_{V}$ (Crane and Duffy, 1998):

$${}^{W}T_{V} = \begin{bmatrix} \cos({}^{W}\gamma_{V}) & -\sin({}^{W}\gamma_{V}) & 0 & {}^{W}x_{V} \\ \sin({}^{W}\gamma_{V}) & \cos({}^{W}\gamma_{V}) & 0 & {}^{W}y_{V} \\ 0 & 0 & 1 & {}^{W}z_{V} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & R_{W} & & T_{W} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$ {}^{V}T_{W} = \begin{bmatrix} & {R_{W}}^{T} & & -{R_{W}}^{T} \cdot T_{W} \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Intersection Navigation

Intersection navigation utilizes path following, intersection detection, and the visual odometer. Figure 3-46 shows a block diagram of the main intersection navigation algorithm. The first process the navigation system does is read in the route instructions specified by the user. This instructs the vehicle how to follow the first path, what to do at the intersection (stop, turn, or go straight), and how to follow the second path. The vehicle proceeds to follow the first path and search for intersections. When it has reached an intersection, it determines if the intersection marks the end of the route, or if it should continue on to navigate through the intersection. When navigating an intersection, the vehicle can either turn or go straight. This instruction is specified in the route instructions. Block diagrams for turning left at a four-way intersection and driving straight through an intersection are shown in Figure 3-47 and Figure 3-48. When the vehicle has successfully navigated through the intersection, it continues following the next path until the next intersection is detected.

The following intersection strategies will be discussed in further detail:

1. Turning at the intersection
2. Going straight through the intersection

A four-way intersection is used to demonstrate these intersection navigation algorithms and is shown in Figure 3-49. The path edges are lined with back tape. Black poster boards mark the corners of the intersections.

Turning Navigation

Ten steps are followed when making a turn at an intersection (see Figure 3-50):

1. Follow initial path
2. Detect intersection
3. Determine where to turn
4. Use visual odometer to guide vehicle
5. Begin turn
6. Track edge of first path
7. Keep turning to next path
8. Track edge of next path
9. Stop turn
10. Follow next path

Step 1: Follow initial path

The vehicle follows its current path using the path following algorithm. Figure 3-51 shows the vehicle following the center of the path.

Step 2: Detect intersection

When an intersection is detected, only the portion of the path below the closest intersection start (marked by the green crosses in Figure 3-52) is used for path following. Path following continues until a start of an intersection passes below the control line (yellow line in Figure 3-52) in the image.

Step 3: Determine where to turn

Figure 3-53 demonstrates this step. When the control line reaches the beginning of an intersection (marked by the green crosses), the vehicle stops following the current path. The last calculated path centerline (indicated by the red line) is recorded. If the vehicle was following an edge, the centerline is defined as a line parallel to the edge shifted the distance desired to follow the edge.

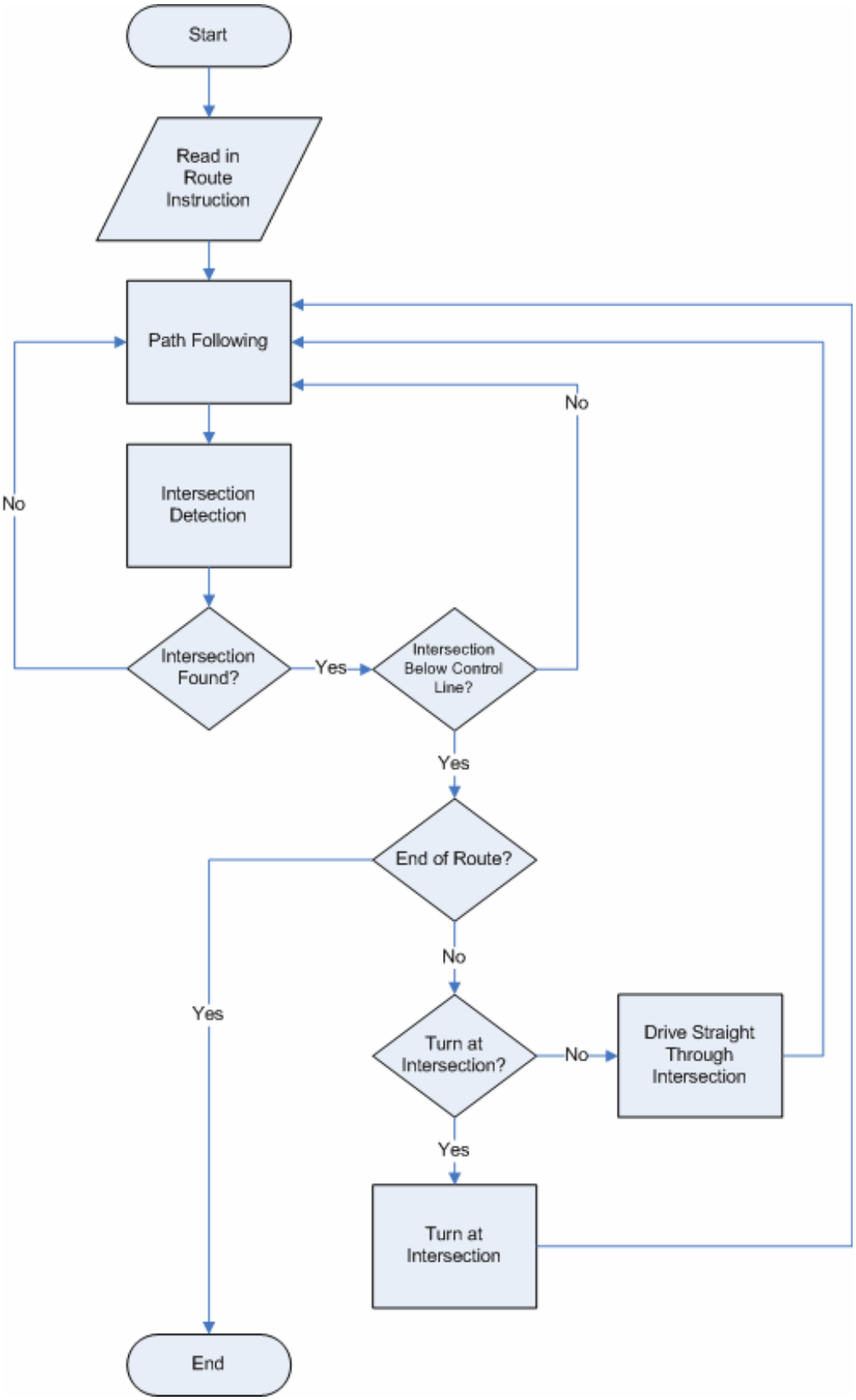


Figure 3-46. Main navigation block diagram.

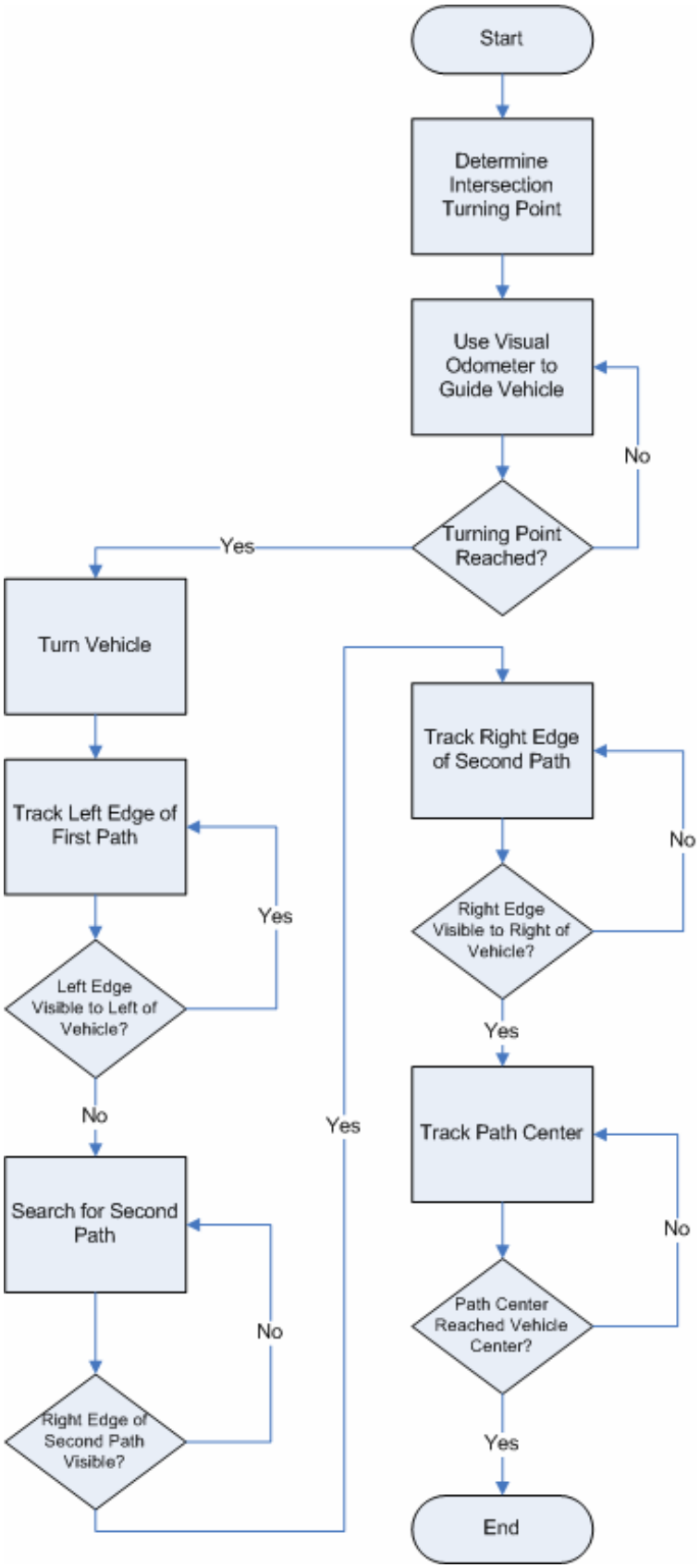


Figure 3-47. Block diagram for making a left turn at a four-way intersection.

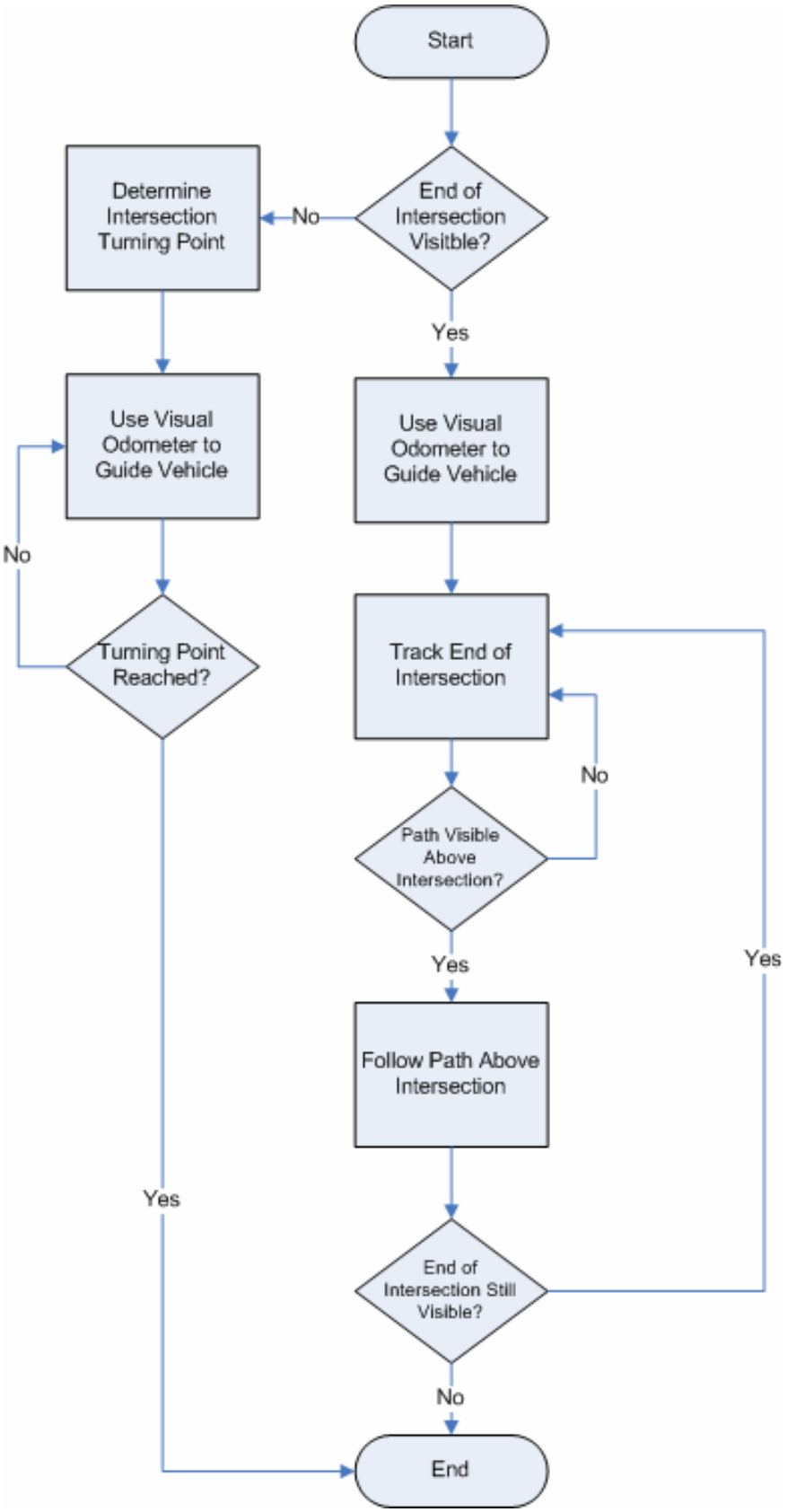


Figure 3-48. Block diagram for driving straight through an intersection.

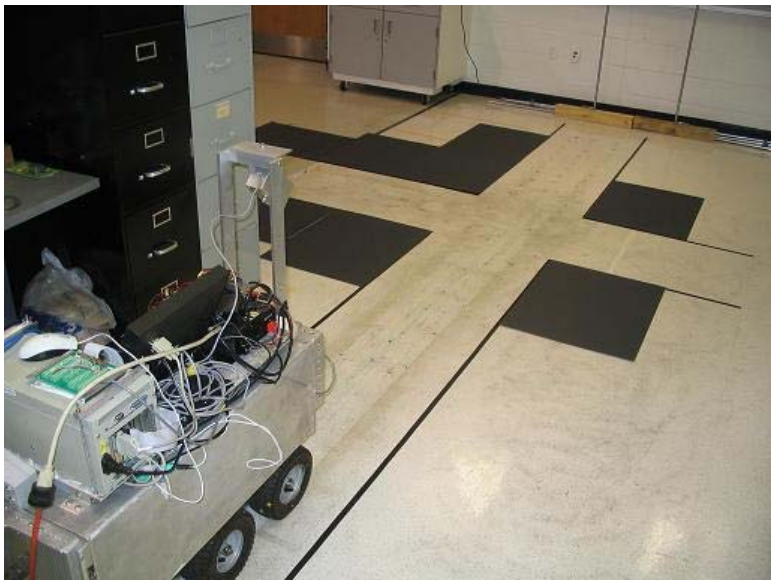


Figure 3-49. Tape intersection.

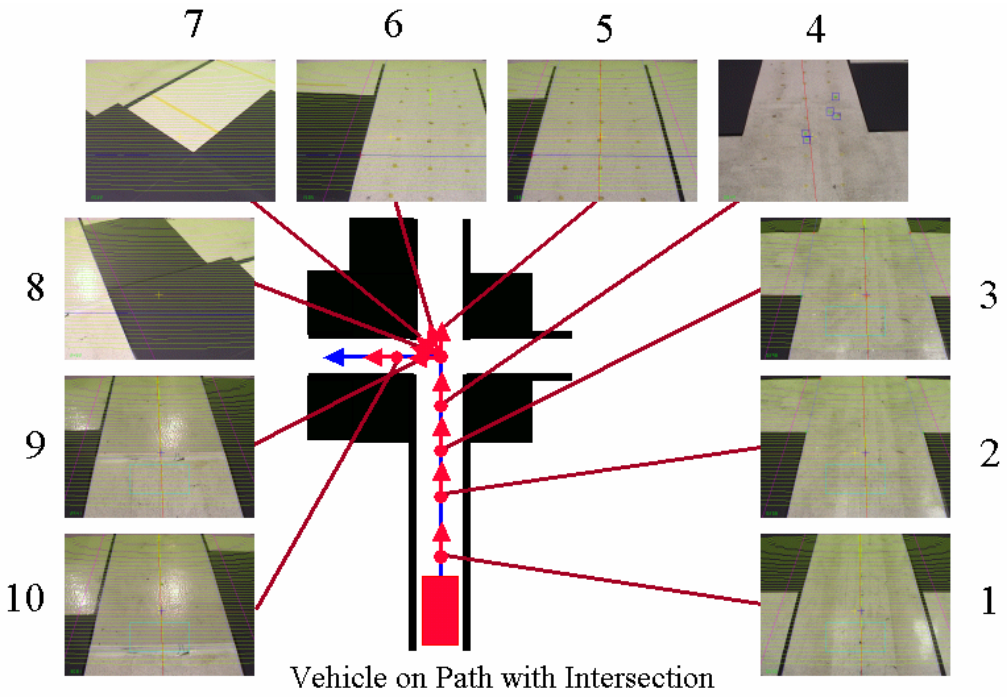


Figure 3-50. Steps for making at turn at an intersection.

The origin of the intersection (marked by the red cross at the beginning of the intersection in Figure 3-53) is defined as the point on the centerline where a line passing through the beginning of the intersection and perpendicular to the centerline crosses. The origin of the intersection is used to define the origin of the world frame in the visual

odometer, with the forward direction of the centerline setting the orientation of the world y-axis.

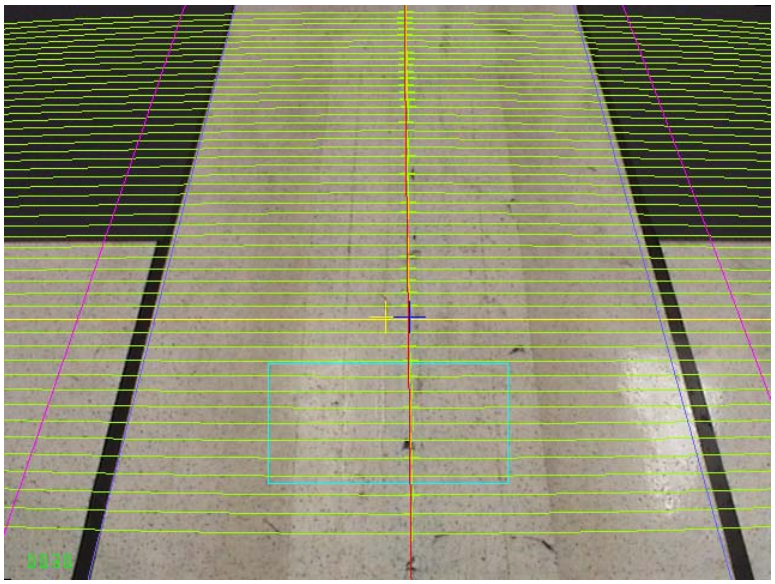

A 
B

Figure 3-51. Turning navigation Step 1: Follow initial path. A) Screen image. B) External view of vehicle on path.

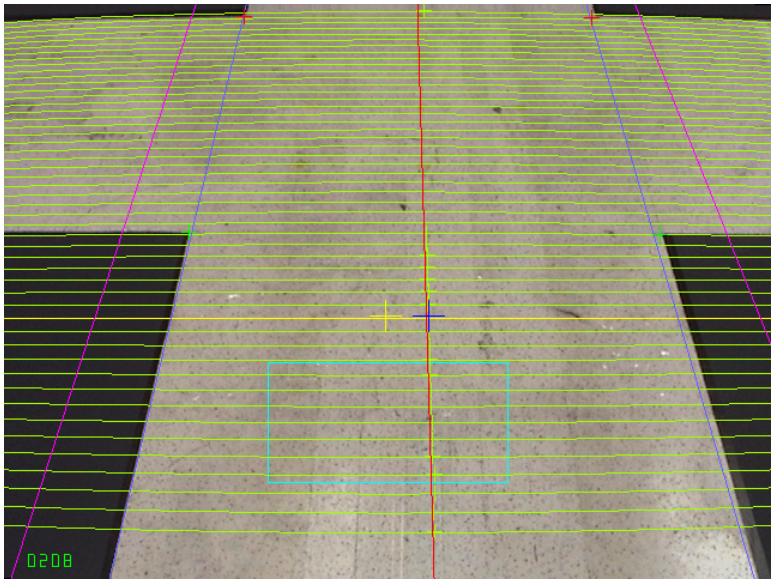

A 
B

Figure 3-52. Turning navigation Step 2: Detect intersection. A) Screen image. B) External view of vehicle on path.

The turquoise cross in Figure 3-53 indicates the desired turning center of the vehicle for the turn. It is placed on the centerline. The greenhouse sprayer being used turns on a point located in the center of the wheels frame, which is its turning center. If

the vehicle is set to follow the center of the intersecting path after the turn, the turning center is placed halfway into the intersection. If the vehicle is set to follow one of the edges of the intersecting path, the turning center is shifted nearer or further into the intersection to acquire the best vehicle position after turning. If the end of the intersection is not visible, the location of the turning center is estimated from a user-defined default width of the intersecting path. The dark blue cross at the top of the image in Figure 3-53 indicates the turning point where the vehicle origin must be to start the turn. It is placed on the centerline and dependent on the location of the turning center.

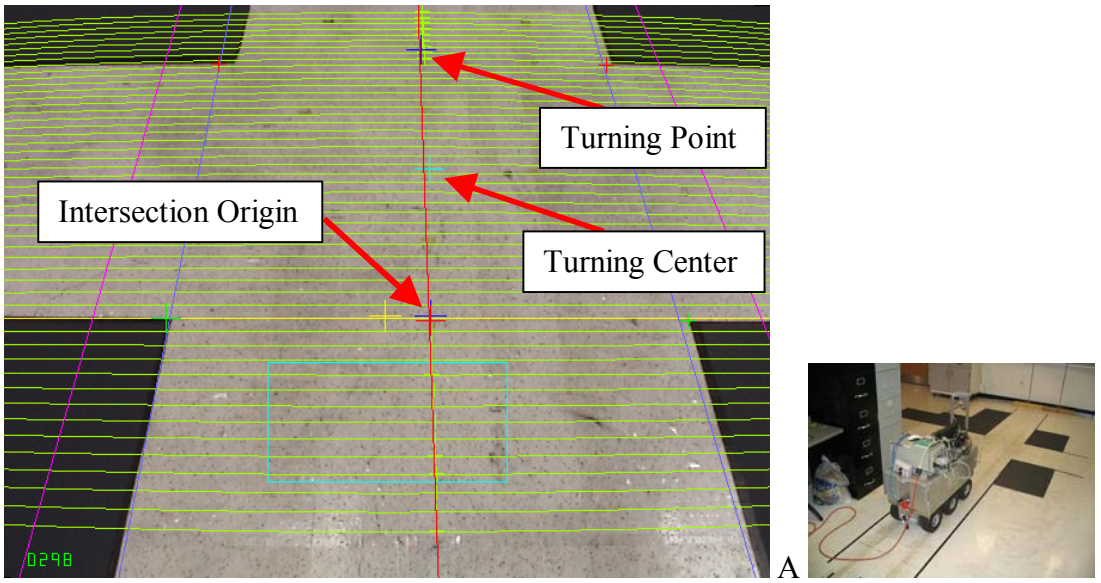


Figure 3-53. Turning navigation Step 3: Determine where to turn. A) Screen image. B) External view of vehicle on path.

Step 4: Use visual odometer to guide vehicle

Features are tracked with the visual odometer as the vehicle drives through the intersection. Based on the translation of the vehicle in the world coordinate system from the visual odometer, the path centerline is shifted and rotated in the vehicle frame. This enables the vehicle to follow a straight line fixed in the world coordinate system using its path following algorithm. The centerline is shifted by taking two arbitrary points (${}^{W}P_{1}$,

${}^{W}P_2$) in the world coordinate system that lie on the original centerline defined in Step 3 and transforming them into the current vehicle coordinate system (${}^{V}P_1, {}^{V}P_2$) using the equations:

$$ {}^{V}P_1 = {}^{V}T_W \cdot {}^{W}P_1 $$

$$ {}^{V}P_2 = {}^{V}T_W \cdot {}^{W}P_2 $$

where ${}^{V}T_W$ is the current transformation matrix relating points in the world frame to the vehicle frame calculated by the visual odometer. A line passing through the transformed points ${}^{V}P_1$ and ${}^{V}P_2$ define the current centerline used for path following. Figure 3-54 shows features being tracked by the odometer and the calculated centerline based on the visual odometer.

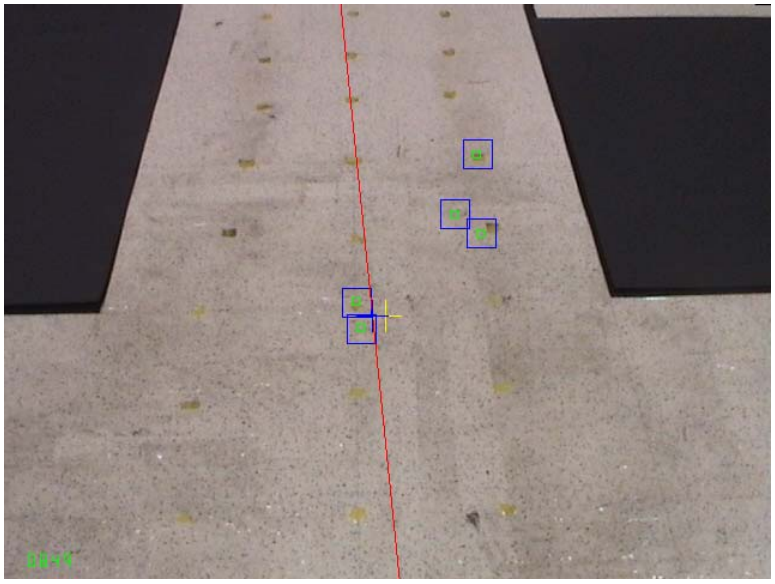
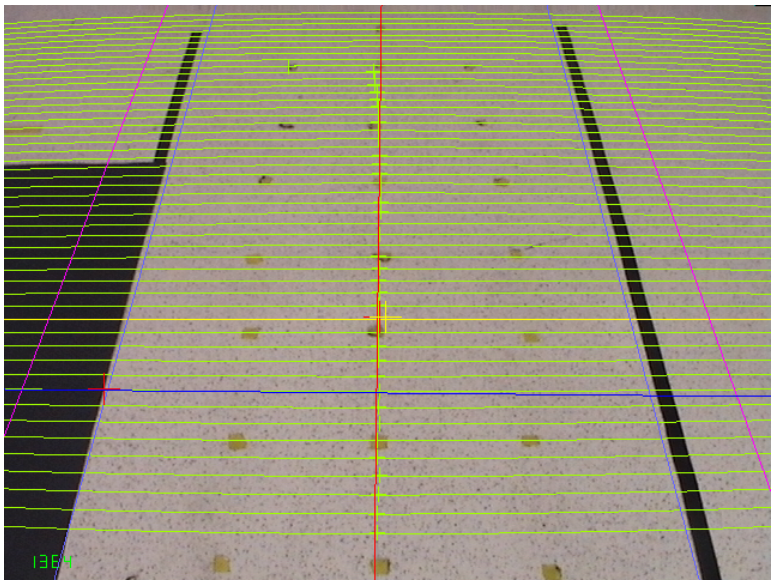

A 
B

Figure 3-54. Turning navigation Step 4: Use visual odometer to guide vehicle. A) Screen image. B) External view of vehicle on path.

Step 5: Begin turn

When the visual odometer indicates that the vehicle has traveled the distance necessary to reach the turning point (calculated in Step 3 and marked dark blue cross at the top of the image in Figure 3-53), the vehicle is instructed to begin turning. While

turning, the path is analyzed to keep track of the first path until it leaves view and the next path comes into view. Figure 3-55 shows the vehicle beginning to turn.



A

B

Figure 3-55. Turning navigation Step 5: Begin turn. A) Screen image. B) External view of vehicle on path.

Step 6: Track edge of first path

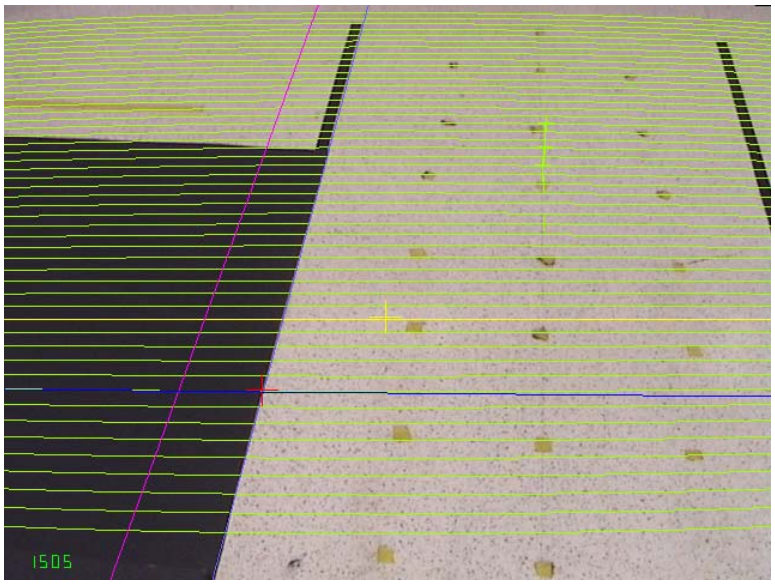
The edge of the first is tracked (marked by the red cross along the blue line in Figure 3-56) during the turn. The location of the edge is defined as the point of intersection between the path edge and the blue line in Figure 3-56. When the edge is visible, the vehicle knows that it is still on the first path. When the edge disappears or passes onto the opposite side the vehicle center (to the right of the yellow cross), the vehicle knows it is facing the intersection corner and can now search for the next path.

Step 7: Keep turning to next path

In Figure 3-57, the vehicle no longer sees the first path and knows to continue turning until the next path is visible.

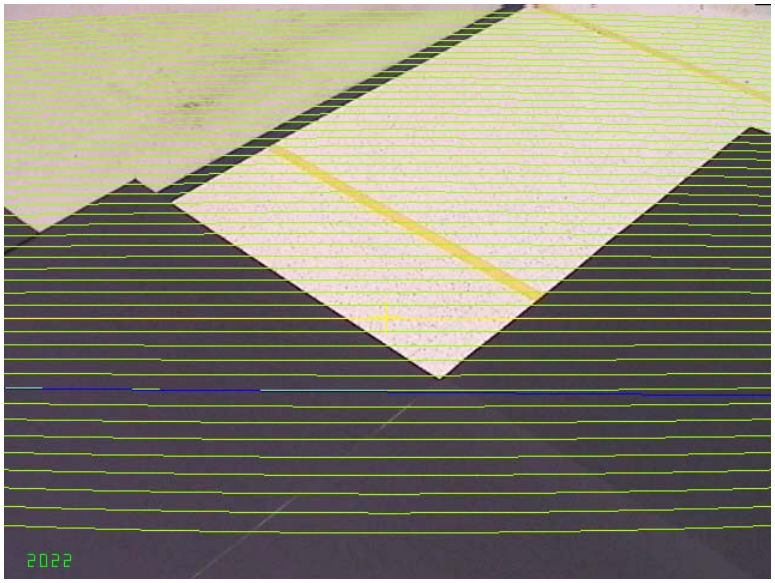
Step 8: Track edge of next path

Once an edge of the next path becomes visible, the vehicle tracks the edges while continuing to turn until the edge reaches the correct side of the vehicle center (to the right of the yellow cross in Figure 3-58). When this occurs, the vehicle knows it is facing the next path.



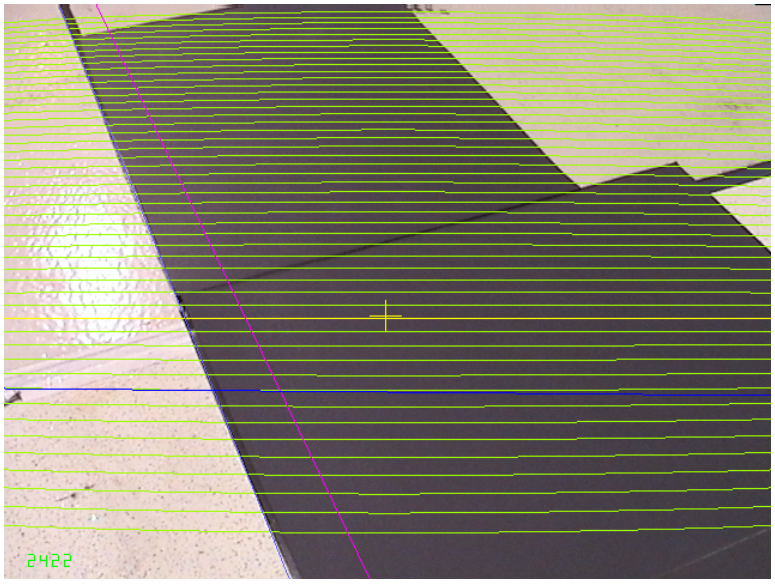
A  B

Figure 3-56. Turning navigation Step 6: Track edge of first path. A) Screen image. B) External view of vehicle on path.



A  B

Figure 3-57. Turning navigation Step 7: Keep turning to next path. A) Screen image. B) External view of vehicle on path.

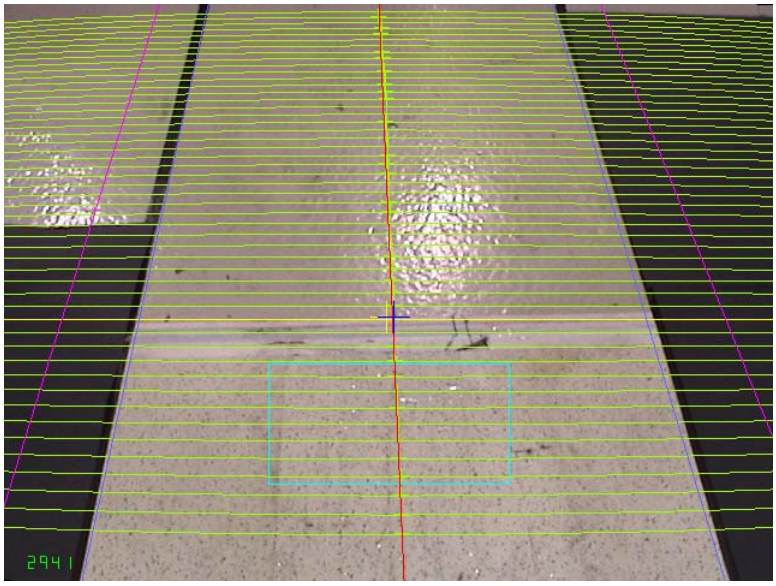


A  B

Figure 3-58. Turning navigation Step 8: Track edge of next path. A) Screen image. B) External view of vehicle on path.

Step 9: Stop turn

The vehicle continues to turn until it is aligned with the centerline or edge it needs to follow. In Figure 3-59, the vehicle stopped turning when the centerline of the next path reached the vehicle center.



A  B

Figure 3-59. Turning navigation Step 9: Stop turn. A) Screen image. B) External view of vehicle on path.

Step 10: Follow next path

The vehicle follows the next path using the path following algorithm. Figure 3-60 shows the vehicle following the center of the path.

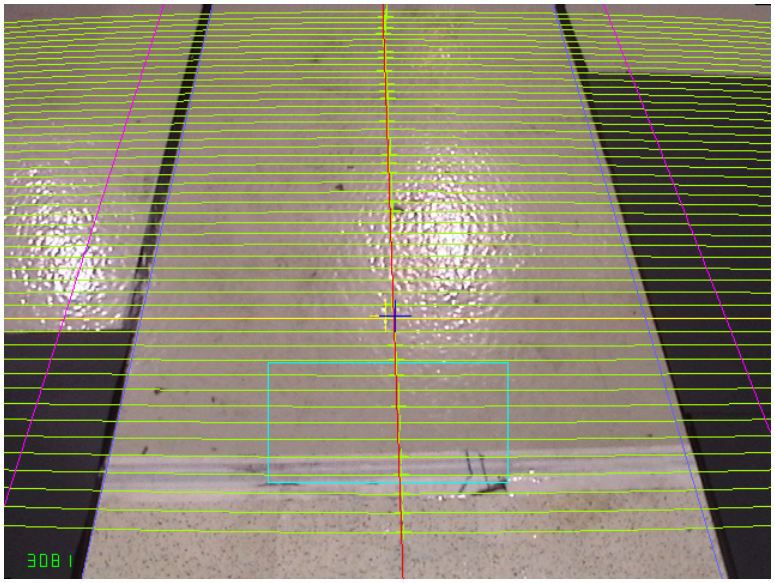

A 
B

Figure 3-60. Turning navigation Step 10: Follow next path. A) Screen image. B) External view of vehicle on path.

Straight Navigation

Five steps are followed driving straight through an intersection (see Figure 3-61):

1. Follow initial path
2. Detect intersection
3. Mark end of intersection
4. Track end of intersection and follow path above intersection
5. Follow next path

Step 1: Follow initial path

The vehicle follows its current path using the path following algorithm. Figure 3-62 shows the vehicle following the center of the path.

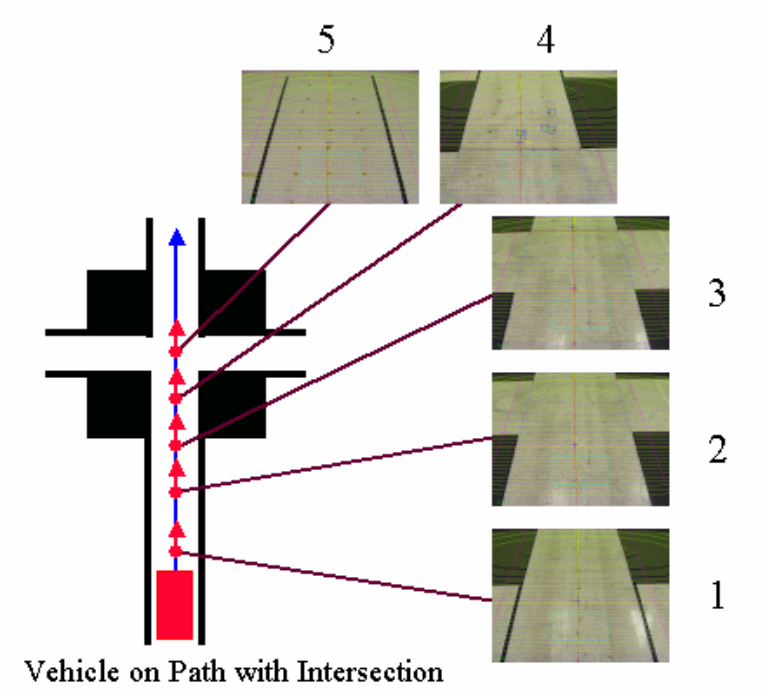
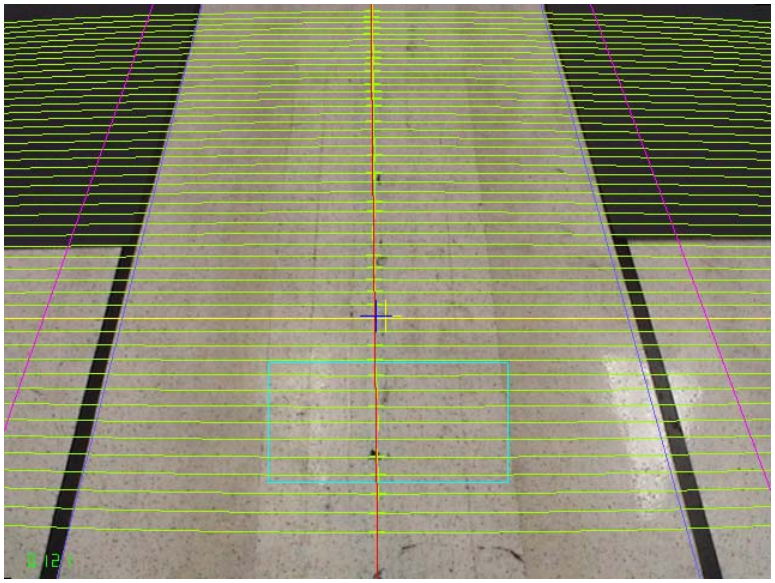


Figure 3-61. Steps for driving straight through an intersection.



A

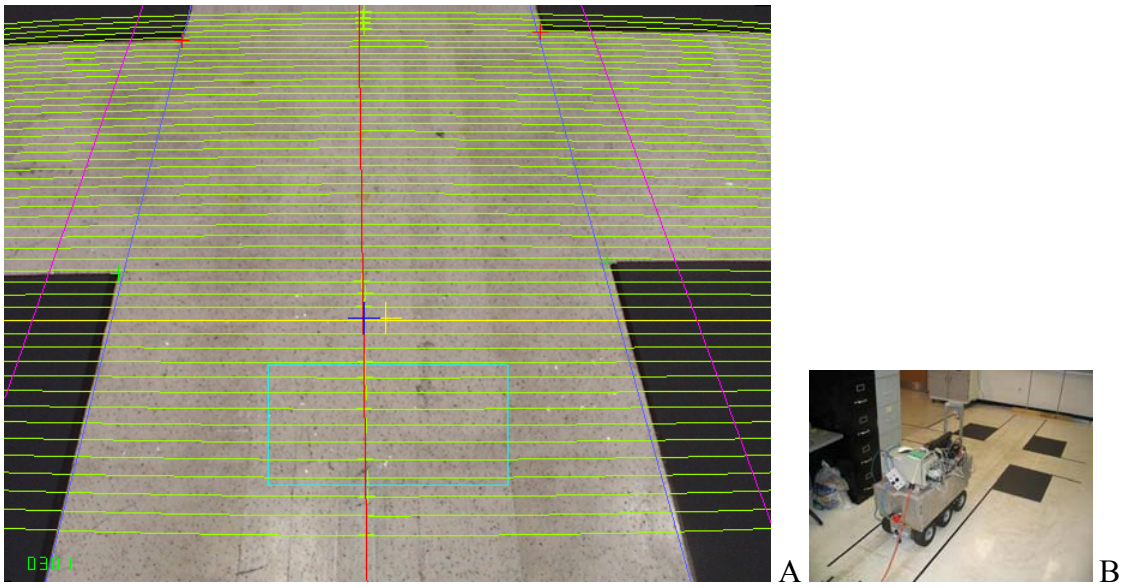
B

Figure 3-62. Straight navigation Step 1: Follow initial path. A) Screen image. B) External view of vehicle on path.

Step 2: Detect intersection

When an intersection is detected, only the portion of the path below the closest intersection start (marked by the green crosses in Figure 3-63) is used for path following.

Path following continues until a start of an intersection passes below the control line (yellow line in Figure 3-63) in the image.



A B

Figure 3-63. Straight navigation Step 2: Detect intersection. A) Screen image. B) External view of vehicle on path.

Step 3: Mark end of intersection

When the control line reaches the beginning of an intersection, the vehicle stops following the current path. If the end of the intersection is detected (marked by the red crosses in Figure 3-64), its distance from the vehicle is recorded. A line perpendicular to the path center and passing through the end of the intersection is drawn on the image (shown as a dark red line in Figure 3-64).

If the end of the intersection is not detected, a turning point is calculated and the visual odometer is used to guide the vehicle as in Steps 3 and 4 of the Turning Navigation algorithm. When the visual odometer determines that the vehicle has reached the turning point, the Straight Navigation algorithm is terminated, and the vehicle begins following the next path.

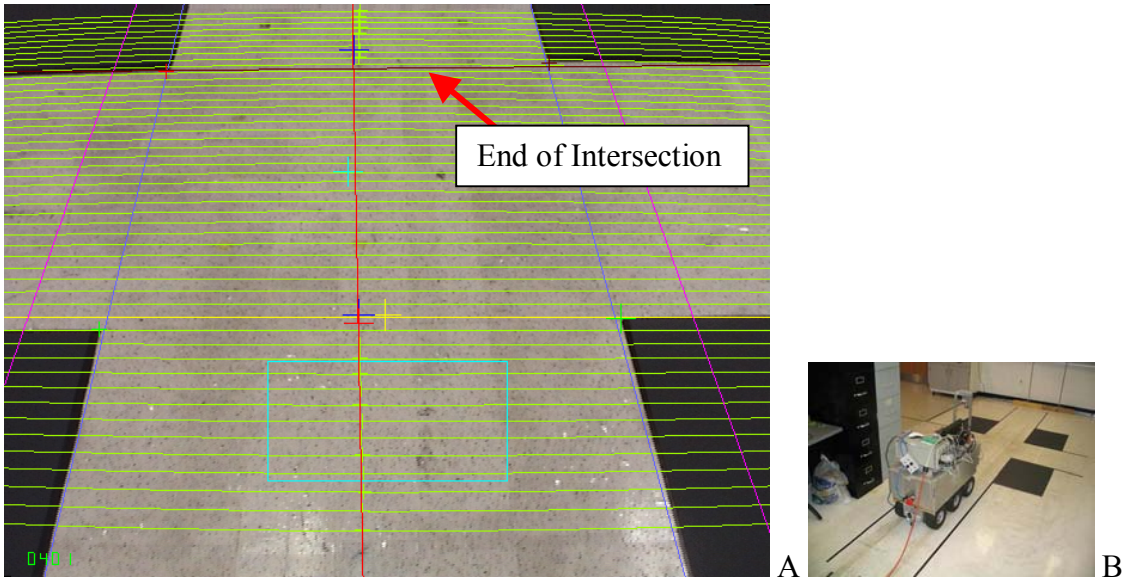


Figure 3-64. Straight navigation Step 3: Mark end of intersection. A) Screen image. B) External view of vehicle on path.

Step 4: Track end of intersection and follow path above intersection

Features are tracked with the visual odometer as the vehicle drives through the intersection. Based on the translation of the vehicle in the world coordinate system from the visual odometer and intersection detection during path analysis, the distance to the end of intersection is computed in the vehicle coordinate system and shifted down the image. If the path above the end of intersection is visible, the vehicle is directed to follow that path. Figure 3-65 shows tracking of the end of the intersection using both the visual odometer and path analysis procedures. Since the next path is visible above the end of the intersection, its centerline is computed and followed.

If the next path beyond the end of the intersection is not visible, the last calculated centerline for the path before the intersection, along with the visual odometer, is used for path following, as described in Steps 4 of the Turning Navigation algorithm, until the next path is visible.

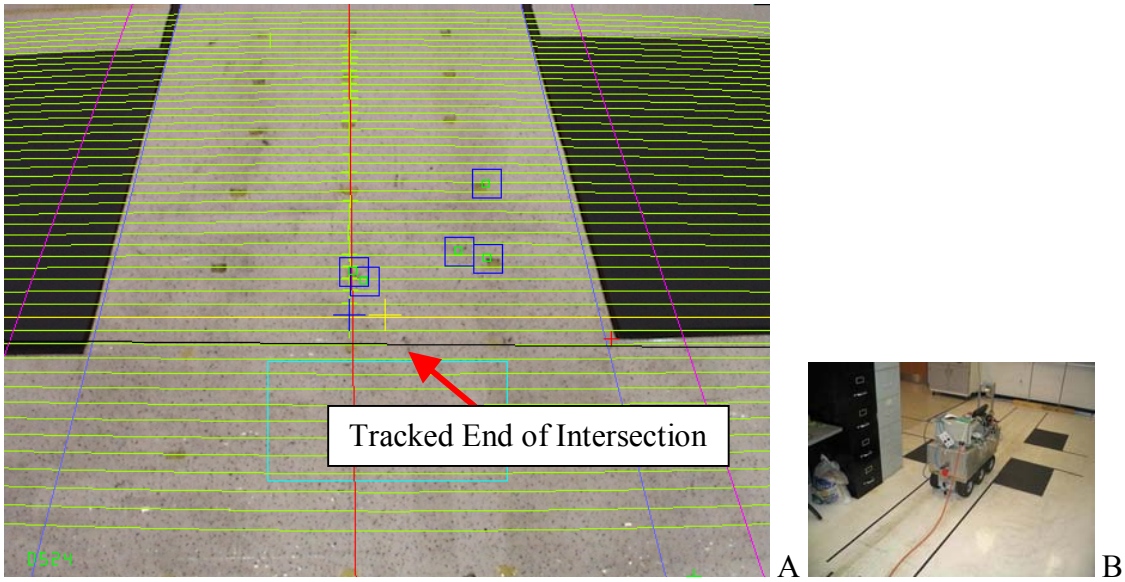


Figure 3-65. Straight navigation Step 4: Track end of intersection and follow path above intersection. A) Screen image. B) External view of vehicle on path.

Step 5: Follow next path

The vehicle follows the next path using the path following algorithm. Figure 3-66 shows the vehicle following the center of the path.

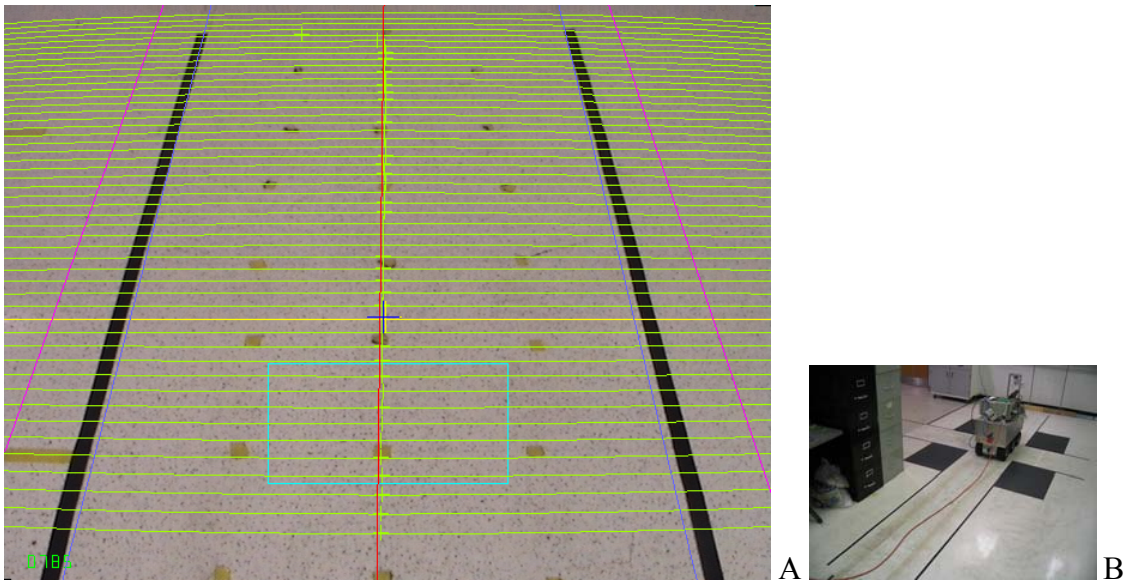


Figure 3-66. Straight navigation Step 5: Follow next path. A) Screen image. B) External view of vehicle on path.

Navigating Other Types of Intersections

If a 180-degree turn is desired, steps 6-8 are repeated before moving onto step 9. If a turn is desired and there is no corner on the intersection as shown in Figure 3-67, steps 6-8 are omitted from the turning algorithm.

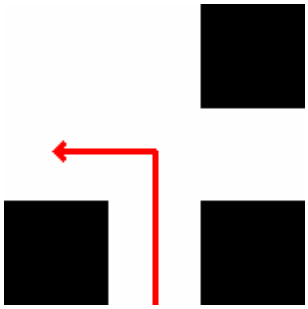


Figure 3-67. Vehicle turn at an intersection without an upper corner.

Route Instruction

A route is broken down into separate segments consisting of paths and intersections. Each route segment consists of a series of specifications used for navigation through that segment. Figure 3-68 shows the route segments used in the 90-degree turn intersection navigation example described above. The route consists of three segments:

1. Path
2. Intersection
3. Path

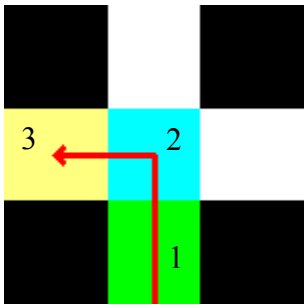


Figure 3-68. Route segments highlighted.

For a "Path" segment, the following items are specified:

- Side of path to follow: C = Center, L = Left, or R = Right
- Type of path: path[i] = TRUE or FALSE, where i = 0 to 2
- Side distance: Distance in inches to stay from the edge (only used when following the left or right edge of the path)
- Path speed: Desired speed to drive motors (range: 0 to 10)
- Path width: Estimate of path width in inches (used to estimated the location of an edge or path centerline if it cannot be found in the image)
- For an "Intersection" segment, the following items are specified
- Type of intersection: inter[i] = TRUE or FALSE, where i = 0 to 5
- Degrees to turn at intersection: -180º to 180º (+ degrees turn right)
- Turn speed: Desired speed to drive motors (range: 0 to 10). For turning, one motor runs at +(Turn speed), while the other runs at –(Turn speed)

To describe the "Type of path" and "Type of intersection", the path and intersection are broken up into cells and are defined by arrays "path" and "inter", as shown in Figure 3-69 and Figure 3-70. If path[i] = TRUE, there is a nonpath element in that cell. If path[i] = FALSE, the cell represents part of the path. inter[i] is defined in a similar fashion. The definitions of each cell are:

path[0] = area to the left of the main path (TRUE if the path has a left edge)

path[1] = main path (always set to FALSE)

path[2] = area to the right of the main path (TRUE if the path has a right edge)

inter[0] = upper-left corner of the intersection

inter[1] = area after intersection (FALSE if a path exists after the intersection)

inter[2] = upper-right corner of the intersection

inter[3] = left of the intersection (FALSE if a left turn is possible)

inter[4] = beginning of the intersection (always set to FALSE)

inter[5] = right of the intersection (FALSE if a right turn is possible)

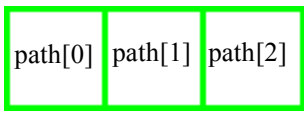


Figure 3-69. Path cells.

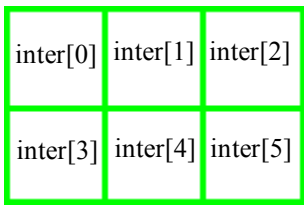


Figure 3-70. Intersection cells.

Figure 3-71 shows an example of how the path and intersection array elements would be assigned for a sample path-intersection-path route.

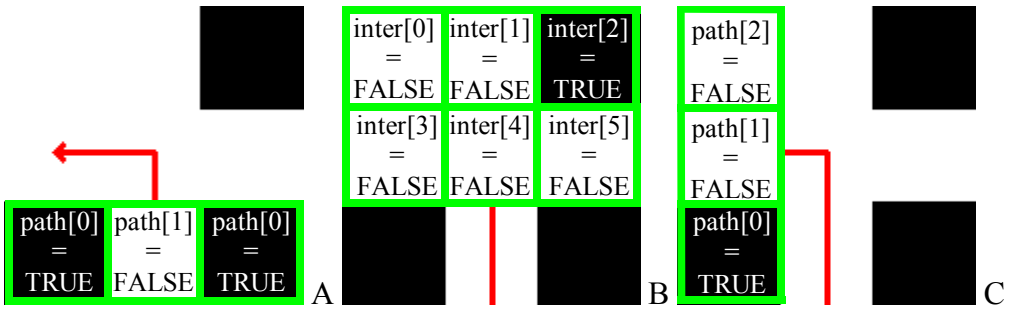


Figure 3-71. Array representation of route segments. A) First path. B) Intersection. C) Next path.

The complete route information input for the previous intersection navigation example for the 90-degree turn in Figure 3-50 is listed below:

```
// Route 0
test_route[0].seg = PATH;
test_route[0].side_follow = C;
test_route[0].path[0] = TRUE;
test_route[0].path[1] = FALSE;
test_route[0].path[2] = TRUE;
test_route[0].side_dist = 12;
test_route[0].path_speed = 3.5;
test_route[0].path_width = 24;

// Route 1
test_route[1].seg = INTER;
test_route[1].inter[0] = TRUE;
test_route[1].inter[1] = FALSE;
test_route[1].inter[2] = TRUE;
test_route[1].inter[3] = FALSE;
test_route[1].inter[4] = FALSE;
test_route[1].inter[5] = FALSE;
test_route[1].deg = -90;
test_route[1].turn_speed = 4;
```

```
// Route 2
test_route[2].seg = PATH;
test_route[2].side_follow = C;
test_route[2].path[0] = TRUE;
test_route[2].path[1] = FALSE;
test_route[2].path[2] = TRUE;
test_route[2].side_dist = 12;
test_route[2].path_speed = 3.5;
test_route[2].path_width = 24;
```

Route 0 lists the instructions for the first path, Route 1 lists the instructions for the intersection, and Route 2 lists the instructions for the second path.

The complete route information input for the previous intersection navigation example for driving straight through the intersection in Figure 3-61 is listed below:

```
// Route 0
test_route[0].seg = PATH;
test_route[0].side_follow = C;
test_route[0].path[0] = TRUE;
test_route[0].path[1] = FALSE;
test_route[0].path[2] = TRUE;
test_route[0].side_dist = 12;
test_route[0].path_speed = 3.5;
test_route[0].path_width = 24;

// Route 1
test_route[1].seg = INTER;
test_route[1].inter[0] = TRUE;
test_route[1].inter[1] = FALSE;
test_route[1].inter[2] = TRUE;
test_route[1].inter[3] = FALSE;
test_route[1].inter[4] = FALSE;
test_route[1].inter[5] = FALSE;
test_route[1].deg = 0;
test_route[1].turn_speed = 4;

// Route 2
test_route[2].seg = PATH;
test_route[2].side_follow = C;
test_route[2].path[0] = TRUE;
test_route[2].path[1] = FALSE;
test_route[2].path[2] = TRUE;
test_route[2].side_dist = 12;
test_route[2].path_speed = 3.5;
test_route[2].path_width = 24;
```

Route 0 lists the instructions for the first path, Route 1 lists the instructions for the intersection, and Route 2 lists the instructions for the second path.

A route can consist of any number of path and intersection segments. However, a path can only be followed by an intersection, and an intersection can only be followed by

a path. An example of a five-segment route consisting of a path, intersection, path, intersection, path is shown in Figure 3-72.

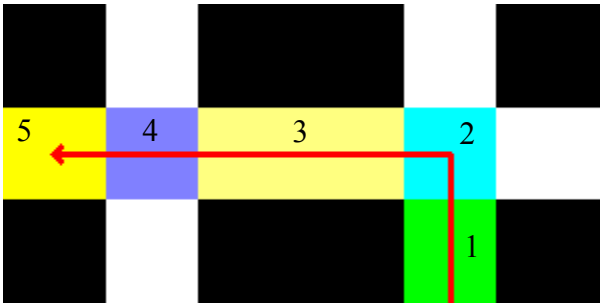


Figure 3-72. Five-segment route.

CHAPTER 4
EXPERIMENTAL METHODS

Several experiments were carried out to verify the following methods utilized for intersection detection and navigation:

- Camera Model
- Visual Odometer
- Intersection Detection
- Intersection Navigation

All experiments were conducted at the University of Florida Agricultural and Biological Engineering Department building. The camera model, intersection detection, and intersection navigation tests were run indoors in a lab setting. The visual odometer tests were completed both indoors in the lab and outdoors on a concrete surface. The vehicle coordinate system as described in Chapter 3 is shown again in Figure 4-1.

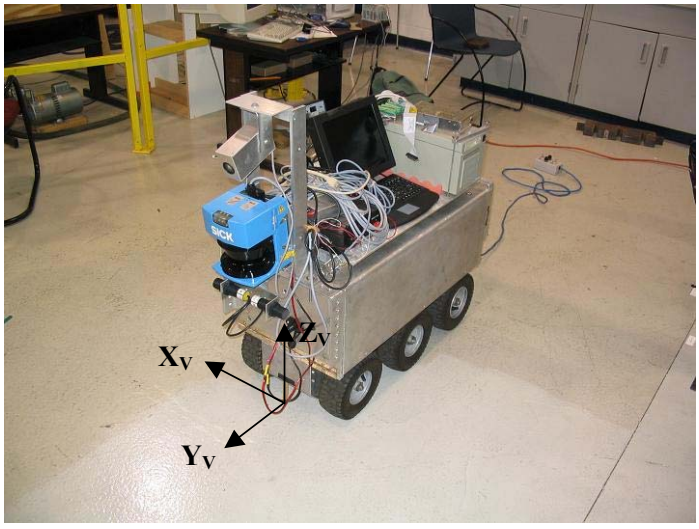
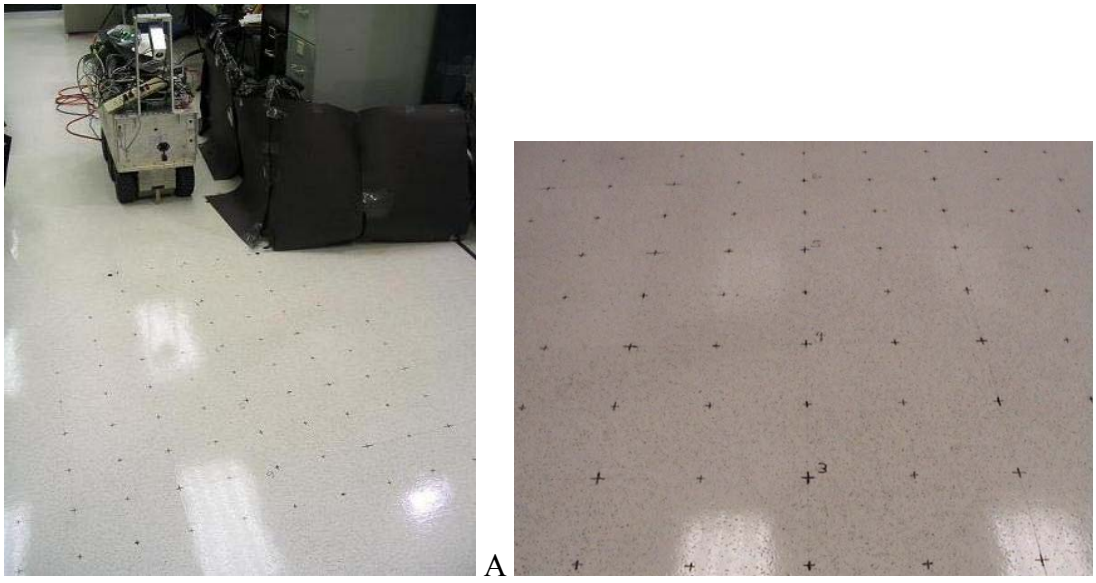


Figure 4-1. Vehicle coordinate system on the autonomous greenhouse sprayer.

Camera Model

The camera model allows interpretation of the digital image acquired from the CCD camera to determine real-world 3D-coordinates. It forms the basis of the visual odometer, path analysis, intersection detection, and intersection navigation algorithms. These algorithms depend on accurate image representation in the camera model for optimal performance. To verify the camera model, grid points spanning the view of the camera were drawn on the ground. Figure 4-2A shows these grid points in front of the vehicle. Each point was spaced out along the vehicle x- and y-coordinate plane in 6-inch increments. The view of these points as seen by the camera is shown in Figure 4-2B. Points along the vehicle y-axis lie down the center of the image. The lower row of points visible in the image is 30 inches from the vehicle. The upper row of points visible in the image is 78 inches from the vehicle.



A B

Figure 4-2. Grid points drawn in front of the vehicle. A) External view. B) Camera image.

A total of sixty-six grid points were visible in the image and used for error analysis. The pixel coordinates of each visible grid point in the camera image in Figure 4-2B were

found by hand and recorded. The 3D coordinates of the grid points were estimated with the camera model for each of these pixel coordinates using the transformation described in Chapter 3: "Transforming Pixels in Flashbus Pixel Plane to 3D Points in Vehicle System." These estimated coordinates from the camera model, $(x_{calc}, y_{calc}, z_{calc})$, were compared to the actual coordinates, $(x_{act}, y_{act}, z_{act})$, of each grid point as measured in the vehicle coordinate system. All coordinates were represented in inches. For each point, an error was calculated:

$$error = \sqrt{(x_{calc} - x_{act})^2 + (y_{calc} - y_{act})^2 + (z_{calc} - z_{act})^2}$$

This error represented how close the camera model was at estimating the 3D-coordinates of that point in the image. Minimum, maximum, and average errors were reported for the set of grid points and will be discussed in the next chapter.

Visual Odometer

The visual odometer estimates vehicle position and orientation over time by tracking ground features. It is utilized during intersection navigation to guide the vehicle and estimate when the vehicle has reached a position in the intersection suitable for turning. Accurate position and orientation estimates are necessary for stable vehicle control and accurate intersection navigation. Three experiments were run to test the visual odometer accuracy and its utilization on various surfaces:

- Translation Test
- Rotation Test
- Verification Tests on Various Surfaces

Translation Test

Vehicle translation estimation is used during intersection navigation to determine when the vehicle has traveled far enough into the intersection to start its turn. The position must be accurate to ensure proper alignment of the vehicle with the next path after turning. Since the vehicle is commanded to drive a specified distance forward along a straight line, the odometer translation test is run on a straight line along the vehicle y-axis. Small squares of tape were placed 6-inches apart along the vehicle x- and y-coordinate plane on the ground along the path to provide adequate features for the visual odometer to track (see Figure 4-3). Figure 4-4 shows the tape marks as seen from the camera image.

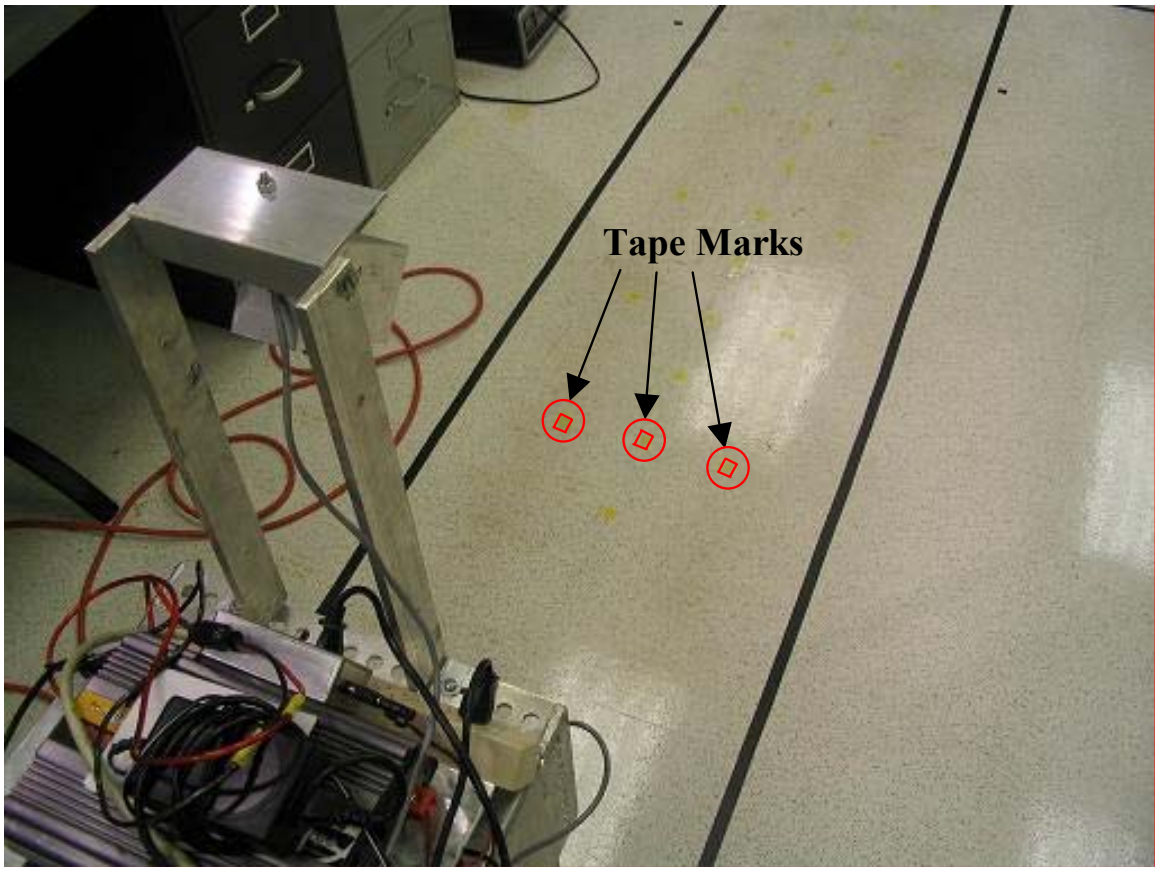


Figure 4-3. Yellow tape marks used for features during odometer test.

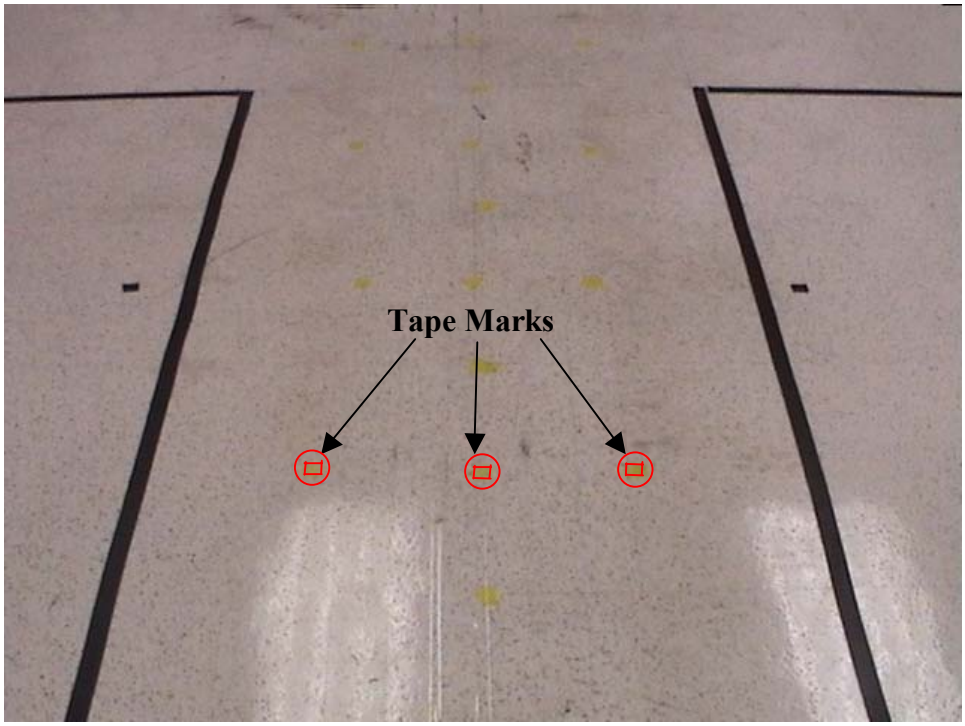


Figure 4-4. Camera view of path with tape marks.

For each test, the vehicle was lined up with a reference line and set up at a starting position, and the world coordinate system was initialized as the starting vehicle coordinate system. The vehicle was instructed to drive straight for a specified distance along the world y-axis direction and stop when the visual odometer read a distance greater than that distance. Distances of 12-inches to 120-inches in 12-inch increments were tested. The largest distance, 120-inches, was selected as the maximum range a vehicle would have to travel to reach the end of a 60-inch wide intersection, if the visual odometer was started when the beginning of the intersection was 45-inches from the vehicle. Three runs were performed for each distance. The vehicle was driven forward at approximately 4.4 inches per second.

The x and y translations from the visual odometer were recorded at the end of each run. The origin of the vehicle coordinate system at the end was marked and its translation in the x- and y-directions were measured relative to the world coordinate

system at the start. Measurements were made to the nearest 1/8-inch using a ruler. The measured vehicle translation was compared to the odometer estimation and an error was calculated.

Rotation Test

A similar test to the translation test was conducted for rotation. Instead of tape marks on the floor, paper marks were used. The marks were placed in a circle approximately 60-inches in front of the vehicle so they only appear in the portion of the image where new features are searched for by the visual odometer (see Figure 4-5).

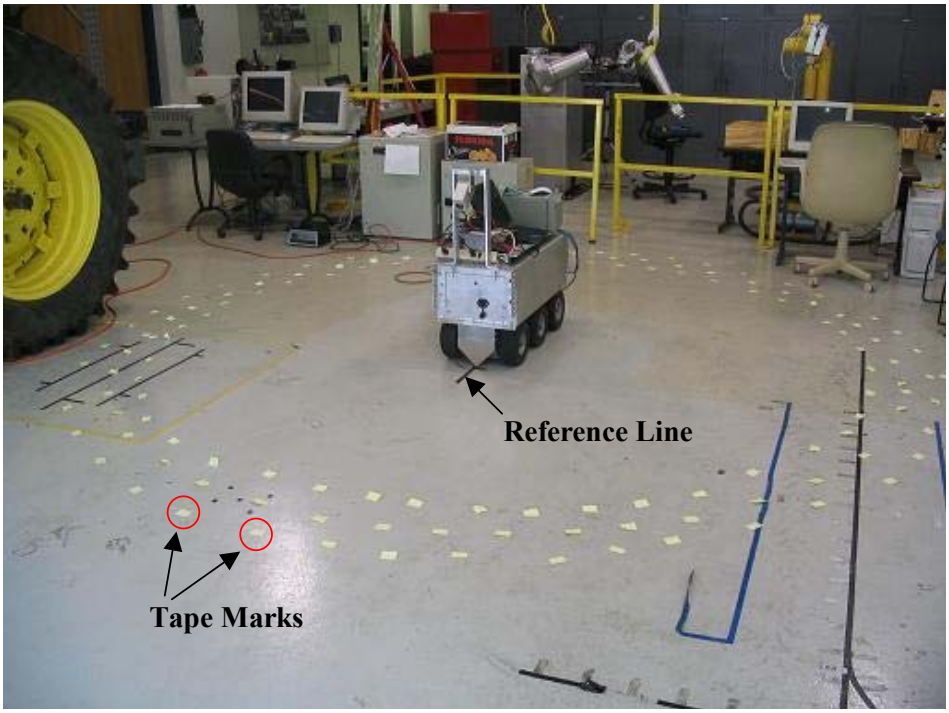


Figure 4-5. Experimental setup for visual odometer rotation test.

For each test, the vehicle was lined up with a reference line (shown in Figure 4-5) and set up at a starting position. The world coordinate system was initialized as the starting vehicle coordinate system. The vehicle was instructed to rotate a specified angle clockwise about the vehicle z-axis and stop when the visual odometer read a distance equal to or greater than that angle. Angles of 45° to 180° in 45° increments were used.

Three runs were performed for each angle. The vehicle was rotated clockwise at approximately 3 degrees per second.

The angle of rotation from the visual odometer was recorded at the end of each run. The front and back of the vehicle were marked on the ground, and a line was drawn between the two to measure vehicle rotation relative to the starting reference line. The angle was measured to the nearest degree. The measured vehicle rotation was compared to the odometer estimation and an error was calculated.

Verification Tests on Various Surfaces

To prove that the visual odometer can work in a greenhouse environment, tests were also run on various surfaces that may be found in a greenhouse. These included concrete, sand, and gravel. All three tests were performed outdoors. The experimental setups for each surface and the view of the surface as seen by the camera are shown in Figure 4-6, Figure 4-7, and Figure 4-8. Sand and gravel were placed along the driving path with a width large enough to cover the area of the image where feature tracking occurs. The material was placed on a tarp for ease of cleanup and did not affect results.


A

B

Figure 4-6. Concrete test setup. A) Concrete surface. B) Camera view.

For each test, the vehicle was lined up with a reference line and set up at a starting position, and the world coordinate system was initialized as the starting vehicle

coordinate system. The vehicle was instructed to drive straight for a 60-inch distance along the world y-axis direction and stop when the visual odometer read a distance greater than that distance. Three runs were made for each surface. The vehicle was driven forward at approximately 4.4 inches per second, as in the translation test.


A 
B

Figure 4-7. Sand test setup. A) Sand surface. B) Camera view.


A 
B

Figure 4-8. Gravel test setup. A) Gravel surface. B) Camera view.

The x and y translations from the visual odometer were recorded at the end of each run. The origin of the vehicle coordinate system at the end was marked, and its translation in the x- and y-directions were measured relative to the vehicle coordinate system at the start. Measurements were made to the nearest 1/8-inch using a ruler. A string was laid down the center of the path along the initial vehicle y-axis at the end of each run as a reference to measure translation (see Figure 4-9). The measured vehicle translation was compared to the odometer estimation and an error was calculated.

Figure 4-9. Reference string laid out after each run for vehicle translation measurement.

Intersection Detection

Intersection detection is one of the primary focuses of this study. Accurate detection of the beginning and end of the intersection is critical for the intersection navigation strategy. Tests were run on a flat intersection lined with tape and a plant intersection lined with potted plants. For each type of intersection, the intersection detection tests were run for both a straight and angled vehicle.

The flat intersection was set up using black tape along the path edges. Black poster board was laid over the four corners of the intersection for stronger corner definition. Figure 4-10 shows the flat intersection setup. Both the main path and intersecting path were 24 inches wide.

The plant intersection was set up using potted plants outside the path edges. For variety, a mix of spartina and iris plants two feet tall was used. Figure 4-11 shows the plant intersection setup. Both the main path and intersecting path were 24 inches wide.

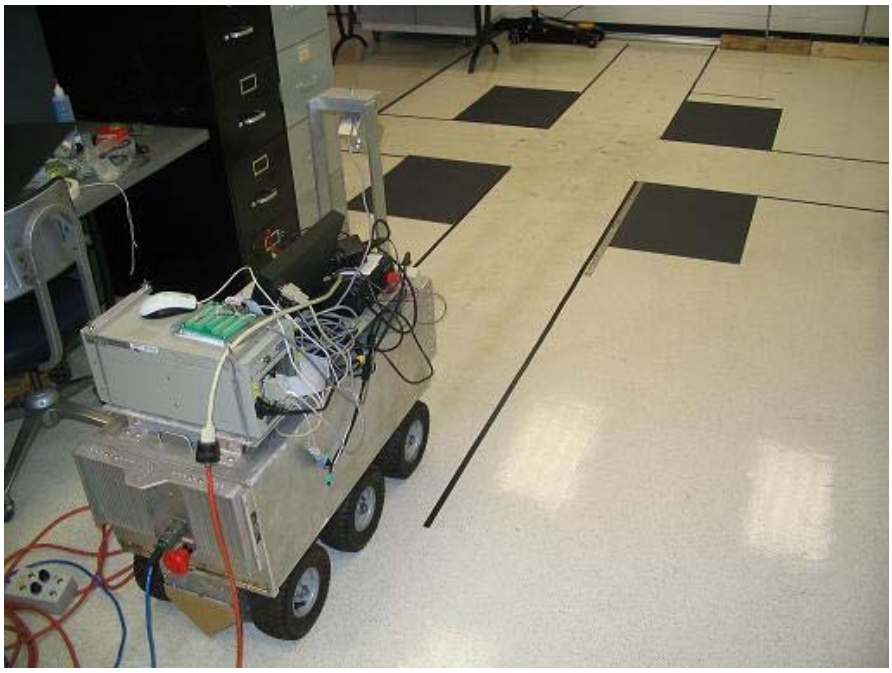


Figure 4-10. Flat intersection setup for intersection detection.

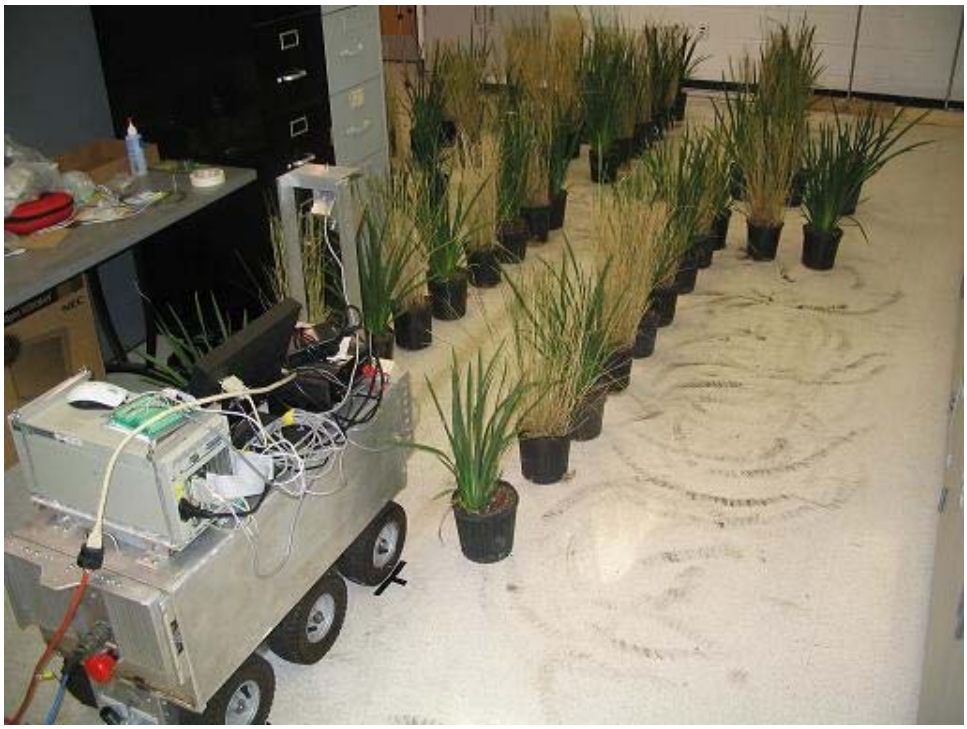


Figure 4-11. Plant intersection setup for intersection detection.

The adaptive thresholding algorithm for path segmentation described in the image processing section of Chapter 3 was disabled during experimentation to provide the same path model for all test images. Since the path in the laboratory was white, a single rgb

threshold value of (100,100,100) was used. Any pixels with all red, green, and blue intensities above 100 were classified as path pixels during the threshold stage of image processing. The intersection detection algorithm was also set to only detect intersections greater than 16 inches wide.

Intersection Detection during Straight Translation

For the straight translation experiment, the vehicle was aligned with the center of the path facing the intersection. Intersection detection tests were run with the origin of the vehicle set up at various distances from the beginning of the intersection. Distances of 72 inches decreasing to 36 inches in 6-inch increments were marked on the path. The vehicle was placed at each of these marked positions, lined up with the path center, and the intersection detection algorithm was run. At the farthest distance, 72 inches, the beginning of the intersection was visible at the top of the image. At the nearest distance, 36 inches, the beginning of the intersection was visible near the bottom of the image. Figure 4-10 shows the vehicle set up 72 inches from the beginning of the intersection. Three runs were performed for each distance.

The locations of the beginning and end of the intersections for both the left and right side of the path found through intersection detection were recorded at the end of each run. Figure 4-12 shows the computed beginning and end of intersection points calculated by the intersection detection program. Note that all points were assumed to lie on the ground plane ($z_v = 0$). False positives (false detection of intersections) and false negatives (non-detection of intersections) were also recorded. All coordinates were recorded in inches and described relative to the current vehicle coordinate system. The detected beginning and end of intersection coordinates were compared to the actual beginning and end of intersection coordinates calculated in the current vehicle coordinate

system. Errors measuring the distance between each of the actual and detected points were calculated and reported.

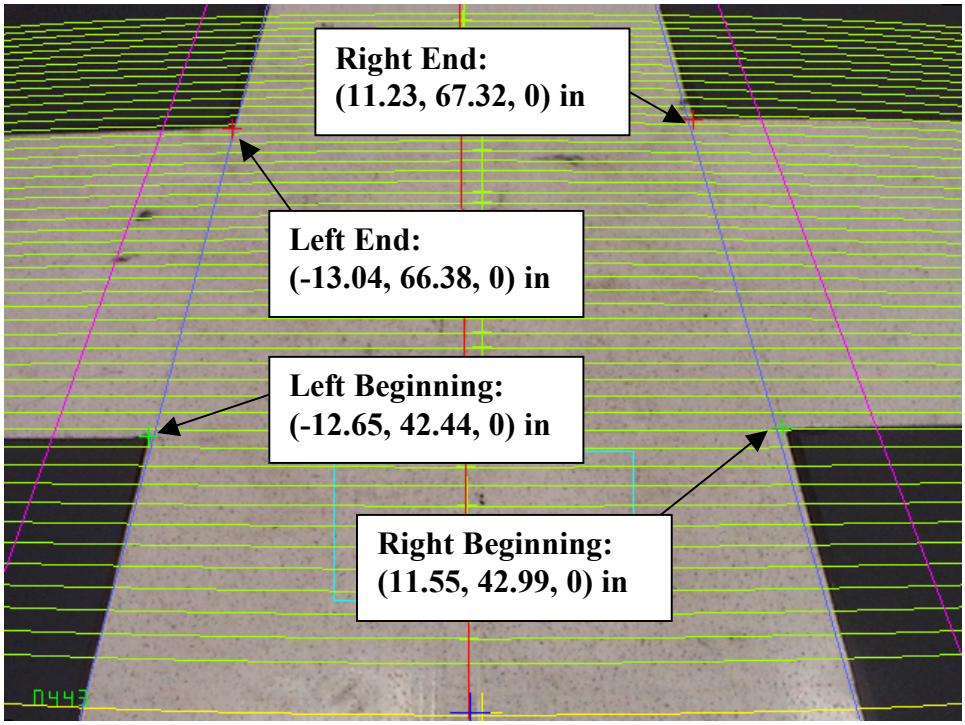


Figure 4-12. Detected intersection points.

Intersection Detection during Rotation

For the rotation experiment, the vehicle was aligned with the center of the path, 48 inches from the beginning of the intersection. Forty-eight inches was chosen because it provided a full view of the intersection. Intersection detection tests were run by rotating the vehicle from –8° to 8° in increments of 2° about its z-axis. Positive angles represent a clockwise rotation of the vehicle. The vehicle was rotated manually to the desired angle, and the intersection detection algorithm was run. Figure 4-13 shows the vehicle set up at an 8° angle, along with the corresponding intersection detection image. Three runs were performed for each angle. Error calculation for the beginning and end of detected intersections was carried out, as in the straight translation experiment.

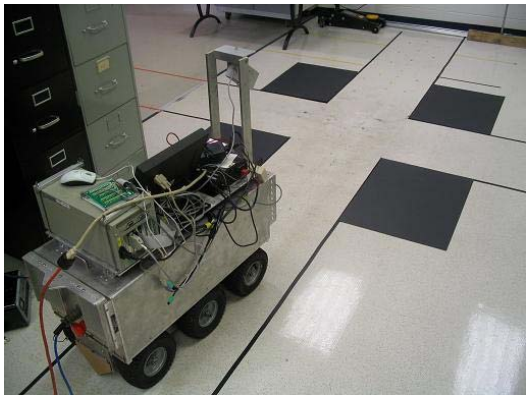
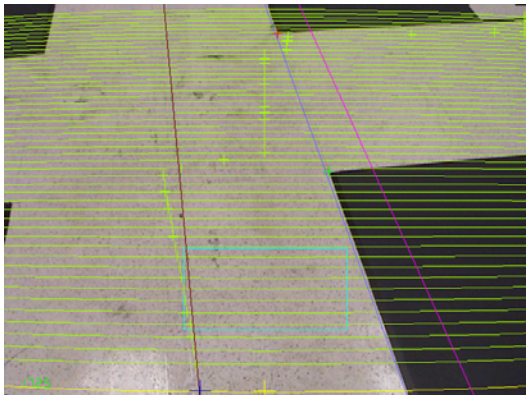
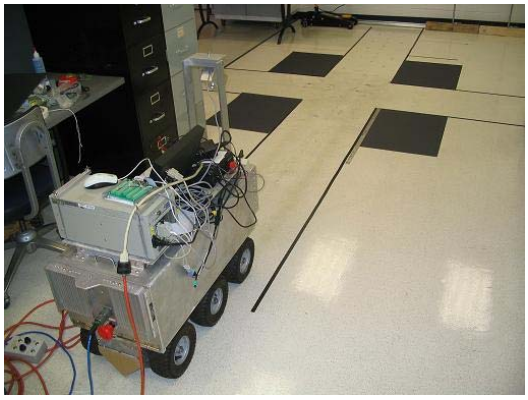

A

B

Figure 4-13. Vehicle rotated at an 8° angle. A) View of vehicle on path. B) Intersection detection image.

Intersection Navigation

Intersection navigation is the primary goal for this study. It depends on successful implementation of all the previously tested algorithms. Tests were run on a flat intersection lined with tape and a plant intersection lined with potted plants. For each type of intersection, the vehicle was instructed to 1) make a left turn, and 2) go straight through the intersection.

Similar intersections with 24-inch wide paths were constructed for the tape and plant intersections as shown in Figure 4-14 and Figure 4-15. For the flat intersection turn setup, additional poster boards were place at the intersection as shown in Figure 4-14B to cover any path-looking area that is seen by the vehicle during the turn.


A

B

Figure 4-14. Flat intersection for intersection navigation. A) Straight setup. B) Turn setup.

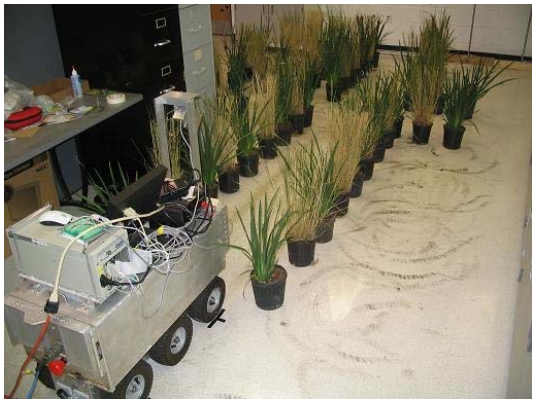


Figure 4-15. Plant intersection for intersection navigation for both straight and turn setups.

For each test, the vehicle origin was lined up with the center of the main path and positioned 96 inches from the beginning of the intersection. The vehicle was aligned with tape markings positioned on the floor to ensure proper vehicle placement at the beginning of each test as seen in Figure 4-16. At the starting position, the vehicle coordinate system is aligned with the world coordinate system. All vehicle measurements along the path were measured relative to this world system. To track vehicle position down the paths and through the intersection, markers were attached to the front and back of the vehicle as shown in Figure 4-17. A blue marker was used for the front, and a green marker was used for the back.

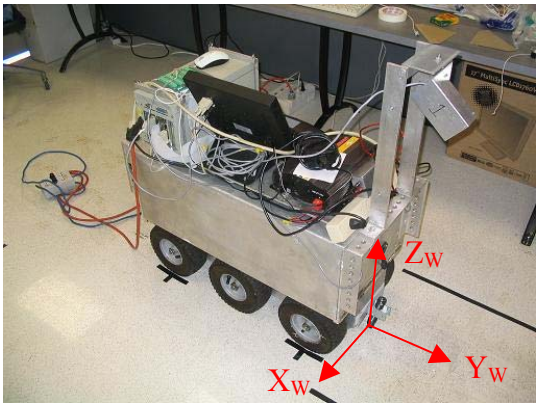


Figure 4-16. Starting position of vehicle for intersection navigation experiments.


A


B

Figure 4-17. Marking devices to track vehicle position. A) Back marker. B) Front marker.

Turning Navigation Experiments

The following instructions were given to the navigation system to complete a left turn:

1. Drive down the center of the first path with a command speed of 3.5 (35% of full motor speed).
2. Turn –90° at the intersection with a turn speed of 4.0 (drive left motor in reverse at 40% of full speed, drive the right motor forward at 40% of full).
3. Drive down the center of the second path with a command speed of 3.5 (35% of full motor speed).

A command speed of 3.5 drove the vehicle at approximately 7.7 inches per second. A turn speed of 4.0 rotated the vehicle at approximately 10 degrees per second. The route information input into the navigation software for these instructions is shown below (see “Route Instruction” in Chapter 3 for further explanation of these terms):

```
// Route 0
test_route[0].seg = PATH;
test_route[0].side_follow = C;
test_route[0].path[0] = TRUE;
test_route[0].path[1] = FALSE;
test_route[0].path[2] = TRUE;
test_route[0].side_dist = 12;
test_route[0].path_speed = 3.5;
test_route[0].path_width = 24;

// Route 1
test_route[1].seg = INTER;
test_route[1].inter[0] = TRUE;
test_route[1].inter[1] = FALSE;
test_route[1].inter[2] = TRUE;
test_route[1].inter[3] = FALSE;
test_route[1].inter[4] = FALSE;
test_route[1].inter[5] = FALSE;
test_route[1].deg = -90;
test_route[1].turn_speed = 4;

// Route 2
test_route[2].seg = PATH;
test_route[2].side_follow = C;
test_route[2].path[0] = TRUE;
test_route[2].path[1] = FALSE;
test_route[2].path[2] = TRUE;
test_route[2].side_dist = 12;
test_route[2].path_speed = 3.5;
test_route[2].path_width = 24;
```

After running the turn routine, the vehicle path drawn by the markers on the ground was recorded. Measurements were made to the nearest ¼-inch in 1-inch increments along the centers of the first path and second (intersecting) path. Path errors relative to the center of the path were calculated for both the vehicle traveling along the first path and the second path. Errors were also calculated to describe how close the front, back, and turning center of the vehicle were to target positions going into and out of the turn.

Straight Navigation Experiments

The following instructions were given to the navigation system to drive straight through the intersection:

1. Drive down the center of the first path with a command speed of 3.5 (35% of full motor speed).
2. Turn 0° at the intersection (drive straight).

3. Drive down the center of the second path with a command speed of 3.5 (35% of full motor speed).

A command speed of 3.5 drove the vehicle at approximately 7.7 inches per second. The route information input into the navigation software for these instructions is shown below:

```
// Route 0
test_route[0].seg = PATH;
test_route[0].side_follow = C;
test_route[0].path[0] = TRUE;
test_route[0].path[1] = FALSE;
test_route[0].path[2] = TRUE;
test_route[0].side_dist = 12;
test_route[0].path_speed = 3.5;
test_route[0].path_width = 24;

// Route 1
test_route[1].seg = INTER;
test_route[1].inter[0] = TRUE;
test_route[1].inter[1] = FALSE;
test_route[1].inter[2] = TRUE;
test_route[1].inter[3] = FALSE;
test_route[1].inter[4] = FALSE;
test_route[1].inter[5] = FALSE;
test_route[1].deg = 0;
test_route[1].turn_speed = 4;

// Route 2
test_route[2].seg = PATH;
test_route[2].side_follow = C;
test_route[2].path[0] = TRUE;
test_route[2].path[1] = FALSE;
test_route[2].path[2] = TRUE;
test_route[2].side_dist = 12;
test_route[2].path_speed = 3.5;
test_route[2].path_width = 24;
```

After running the turn routine, the vehicle path drawn by the markers on the ground was recorded. Measurements were made to the nearest ¼-inch in 1-inch increments along the center of the first path, through the center of the intersection, and along the center of the second path. Path errors relative to the center of the path were calculated.

CHAPTER 5
RESULTS

Results were obtained for the following experiments described in the previous chapter:

- Camera Model
- Visual Odometer
- Intersection Detection
- Intersection Navigation

Camera Model

Table 5-1 displays the minimum, maximum, and average errors resulting from the sixty-six grid points tested. The maximum error occurred at a point far away from the vehicle and appeared in the upper-right portion of the image. The minimum error occurred at a point nearer to the vehicle and appeared in the lower-left portion of the image.

Table 5-1. Error results for the camera model verification experiment.

Error Type	Error (in)	Grid Point Coordinates (in)
Maximum	3.70	(24.00, 78.00, 0.00)
Minimum	0.08	(-12.00, 36.00, 0.00)
Average	1.09	–

The error distribution throughout the image can be better visualized by projecting the 3D-coordinates of the grid points into the pixel plane using the camera model as described in Chapter 3: “Transforming 3D Points in Vehicle System to Pixels in Flashbus

Pixel Plane." Figure 5-1 shows the resulting image. The red points in the figure indicate the 3D-coordinates of the grid points projected in the image. The green points in the figure indicate the actual location of the grid points marked in the image by hand. The minimum and maximum error locations reported in Table 5-1 are pointed out in the figure.

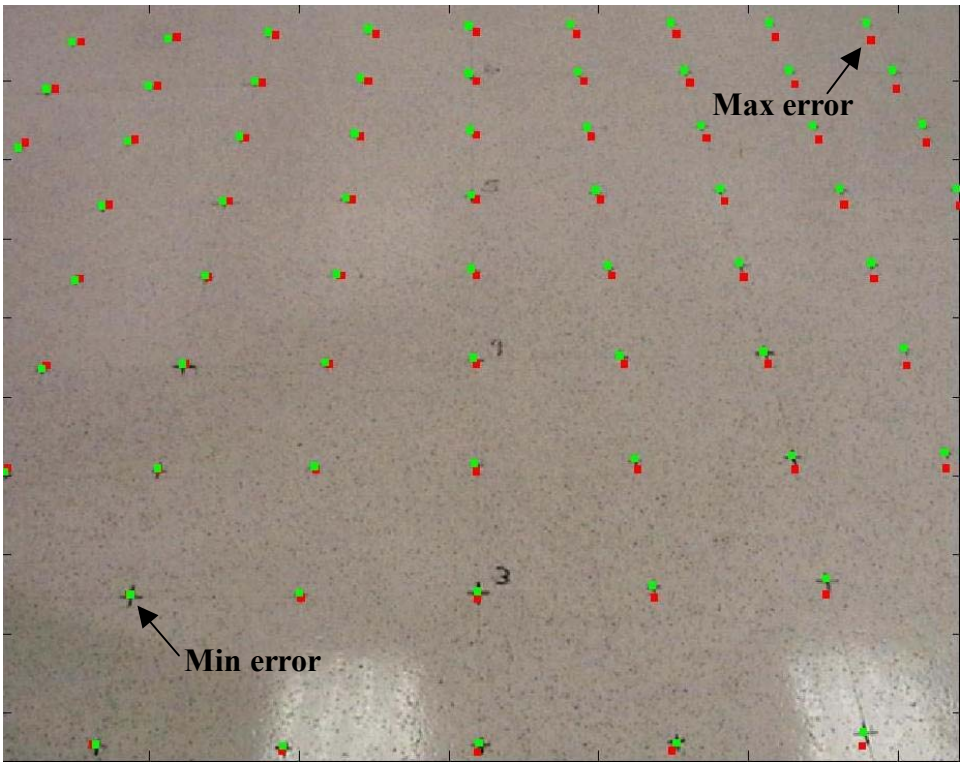


Figure 5-1. Comparison of projected grid points (red) to actual (green) in image.

The results show that the developed camera model related pixels to real-world coordinates with a maximum error of 3.70 inches at a distance 78.00 inches away, and minimum error of 0.08 inches at a distance of 36 inches away. Points seen in the lower-left portion of the image lead to the lowest correspondence errors, while points seen in the upper-right portion of the image lead to the highest correspondence errors. Much of this error was due to the accuracy of the extrinsic camera parameters developed for the camera model discussed in Chapter 3. Slight deviations in the camera geometry from the

perfect model (shown in Figure 3-4) led to small errors reported in the image. Based on the movement of the projected points compared to the actual points in the image, the camera appears to have exhibited a slight rotation about the point with minimum error.

Visual Odometer

Results were obtained from each of the visual odometer tests described in the previous chapter:

- Translation Test
- Rotation Test
- Verification Tests on Various Surfaces

The visual odometer ran at an average of 10 Hz during experimentation.

Translation Test

The results obtained from the three translation test runs are shown in Table 5-2, Table 5-3, and Table 5-4.

Table 5-2. Translation test Run 1.

Command Distance (in)	Odometer x_v (in)	Odometer y_v (in)	Measured x_v (in)	Measured y_v (in)	Error (in)
12	1.66	12.77	-0.25	13.00	1.92
24	0.26	24.76	0.00	25.63	0.90
36	-1.29	36.46	0.13	37.50	1.76
48	-4.30	48.80	0.13	49.00	4.43
60	-1.27	60.73	0.38	61.75	1.94
72	0.49	72.73	-0.63	72.88	1.13
84	2.18	84.89	0.00	84.50	2.21
96	-2.47	96.89	-1.75	95.75	1.35
108	4.73	108.72	-1.50	109.00	6.24
120	-11.61	120.92	-0.63	120.50	10.99

The errors listed in the tables represent the difference in distance between the measured vehicle position and the position estimated by the visual odometer. Table 5-5 shows the average, minimum, and maximum errors from the three runs. The general

error range remained consistent throughout the range of distances tested. These odometer errors are the result of the camera model errors relating pixels to points in the vehicle coordinate system as discussed in the previous experiment. Even though a ground feature may have been tracked successfully from the top of the image to the bottom, its perceived coordinates by the camera model could be off by several inches by the time it reaches the bottom of the image. Depending where in the image a feature was originally found and how far it was tracked down the path determined the error contribution it made.

Table 5-3. Translation test Run 2.

Command Distance (in)	Odometer x_v (in)	Odometer y_v (in)	Measured x_v (in)	Measured y_v (in)	Error (in)
12	0.62	12.79	-0.13	13.25	0.88
24	4.10	25.31	0.13	26.50	4.14
36	0.29	36.88	0.25	38.00	1.12
48	1.41	48.62	0.00	49.50	1.66
60	-1.94	60.76	0.00	60.75	1.94
72	3.69	72.55	-0.50	72.50	4.19
84	-0.41	84.58	-0.75	83.25	1.37
96	-0.12	96.47	-1.38	96.25	1.28
108	-2.64	108.56	-0.25	107.13	2.79
120	0.07	120.45	-1.00	118.38	2.33

Table 5-4. Translation test Run 3.

Command Distance (in)	Odometer x_v (in)	Odometer y_v (in)	Measured x_v (in)	Measured y_v (in)	Error (in)
12	2.86	13.07	0.00	13.75	2.94
24	-2.84	24.79	0.00	25.50	2.93
36	0.33	36.51	0.13	37.50	1.01
48	-0.48	48.66	0.13	49.75	1.24
60	-3.07	60.90	0.13	60.25	3.26
72	1.38	72.92	0.38	71.75	1.54
84	-2.00	84.72	0.38	84.88	2.38
96	-3.74	96.99	-1.38	96.88	2.37
108	2.82	108.65	-1.00	108.13	3.85
120	-0.30	120.89	-1.25	120.00	1.30

Several runs gave very high errors above 4 inches, with the largest occurring during Run 1 at the 120-inch distance with a 10.99-inch error. The main contributions to these errors were poor estimation of the vehicle translation along its x-axis. Reasons for this error have not been determined and appear random.

Table 5-5. Average error over the three translation test runs.

Command Distance (in)	Average Error (in)	Minimum Error (in)	Maximum Error (in)
12	1.91	0.88	2.94
24	2.66	0.90	4.14
36	1.30	1.01	1.76
48	2.45	1.24	4.43
60	2.38	1.94	3.26
72	2.29	1.13	4.19
84	1.99	1.37	2.38
96	1.66	1.28	2.37
108	4.29	2.79	6.24
120	4.88	1.30	10.99

Rotation Test

The results obtained from the three rotation test runs are shown in Table 5-6, Table 5-7, and Table 5-8.

Table 5-6. Rotation test Run 1.

Command Angle (deg)	Odometer Angle (deg)	Measured Angle (deg)	Error (deg)
45	45	45	0
90	90	90	0
135	136	145	9
180	182	190	8

The errors listed in the tables represent the difference in rotation between the measured vehicle orientation and the orientation estimated by the visual odometer. Table 5-9 shows the average, minimum, and maximum errors from the three runs. There is a

general trend that the farther the command distance, the larger the error. The maximum error during experimentation was 20°, which occurred during a 135° turn.

Table 5-7. Rotation test Run 2.

Command Angle (deg)	Odometer Angle (deg)	Measured Angle (deg)	Error (deg)
45	46	48	2
90	90	88	2
135	136	150	14
180	182	166	16

Table 5-8. Rotation test Run 3.

Command Angle (deg)	Odometer Angle (deg)	Measured Angle (deg)	Error (deg)
45	46	44	2
90	90	91	1
135	135	155	20
180	181	182	1

Table 5-9. Average error over the three rotation test runs.

Command Angle (deg)	Average Error (deg)	Minimum Error (deg)	Maximum Error (deg)
45	1	3	2
90	1	2	2
135	14	7	20
180	8	26	16

Verification Tests on Various Surfaces

The results obtained for a commanded vehicle translation of 60 inches on concrete, sand, and gravel are shown in Table 5-10, Table 5-11, and Table 5-12.

Table 5-10. Translation test for concrete.

Run	Odometer x_v (in)	Odometer y_v (in)	Measured x_v (in)	Measured y_v (in)	Error (in)
1	-7.81	60.90	-0.13	59.25	7.86
2	-4.11	60.66	-0.13	59.25	4.23
3	-2.48	60.84	-0.13	59.88	2.54

Table 5-11. Translation test for sand.

Run	Odometer x_v (in)	Odometer y_v (in)	Measured x_v (in)	Measured y_v (in)	Error (in)
1	-2.35	60.64	-1.38	59.38	1.60
2	-2.88	60.62	-1.00	58.50	2.83
3	-2.83	60.72	-1.75	60.00	1.17

Table 5-12. Translation test for gravel.

Run	Odometer x_v (in)	Odometer y_v (in)	Measured x_v (in)	Measured y_v (in)	Error (in)
1	-4.45	60.17	-2.00	57.50	3.62
2	-2.79	60.77	-2.00	58.00	2.88
3	-2.67	60.63	-1.50	57.00	3.81

The errors listed in the tables represent the difference in distance between the measured vehicle position and the position estimated by the visual odometer. Table 5-5 shows the average, minimum, and maximum errors from the three runs for each surface. The errors acquired from the testing on the lab floor at 60 inches as reported in Table 5-5 is also shown in Table 5-13 for comparison. The results show similar accuracy for the visual odometer on all surfaces.

Table 5-13. Average error over the three test runs for concrete, sand, and gravel.

Surface	Average Error (in)	Minimum Error (in)	Maximum Error (in)
Concrete	4.88	2.54	7.86
Sand	1.87	1.17	2.83
Gravel	3.44	2.88	3.81
Lab	2.38	1.94	3.26

Intersection Detection

Results were obtained from experiments on a flat intersection lined with tape and plant intersection lined with potted plants, as described in the previous chapter. For each type of intersection, the intersection detection tests were run for both a straight and angled vehicle. Figure 5-2, Figure 5-3, Figure 5-4, and Figure 5-5 display sample images

showing successful intersection detection for each of these experiments. The green crosses along the blue path edge lines mark the detected beginning of intersections, and the red crosses along the blue path edge lines mark the detected end of intersections. The intersection detection algorithm ran at an average of 5 Hz during experimentation.

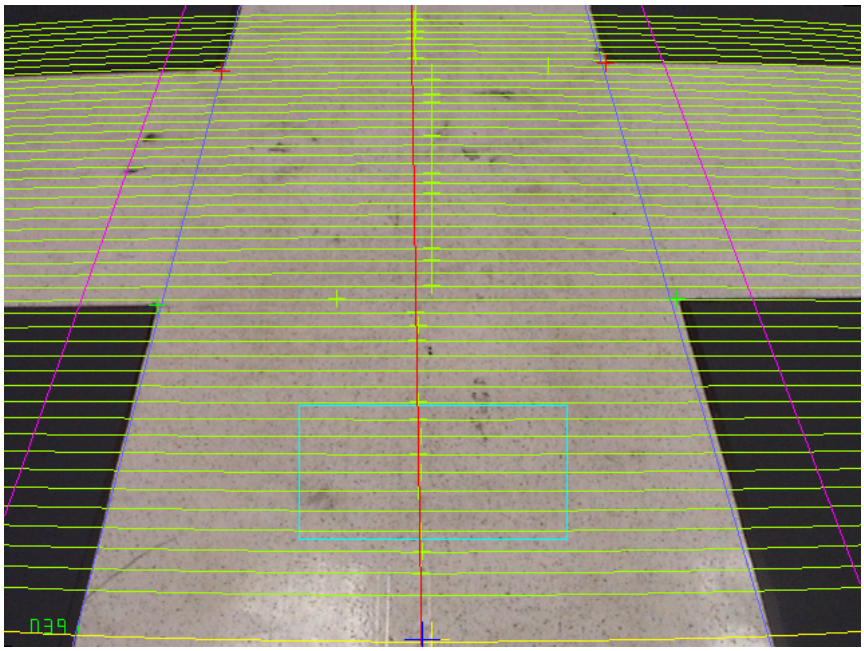


Figure 5-2. Intersection detection image from straight translation for flat intersection Run 1. Vehicle positioned 48 inches from intersection.

Intersection Detection during Straight Translation

The following are intersection detection results for both the flat intersection and plant intersection setups.

Straight translation: Flat intersection

Figure 5-6 shows the intersection detection images taken over a range of distances from Run 1 of the straight translation for flat intersection experiment.

The results from the three intersection detection runs of the flat intersection by translating the vehicle along the center of the path are reported in Table 5-14, Table 5-15, and Table 5-16. Errors are reported for how accurate the coordinates of the beginning

and end of intersections on the left and right edges of the path were found using the intersection detection algorithm. "FN" stands for false negative (the intersection point was visible in the image but not found). "NV" stands for not visible (the intersection point was not visible in the image at that particular distance).

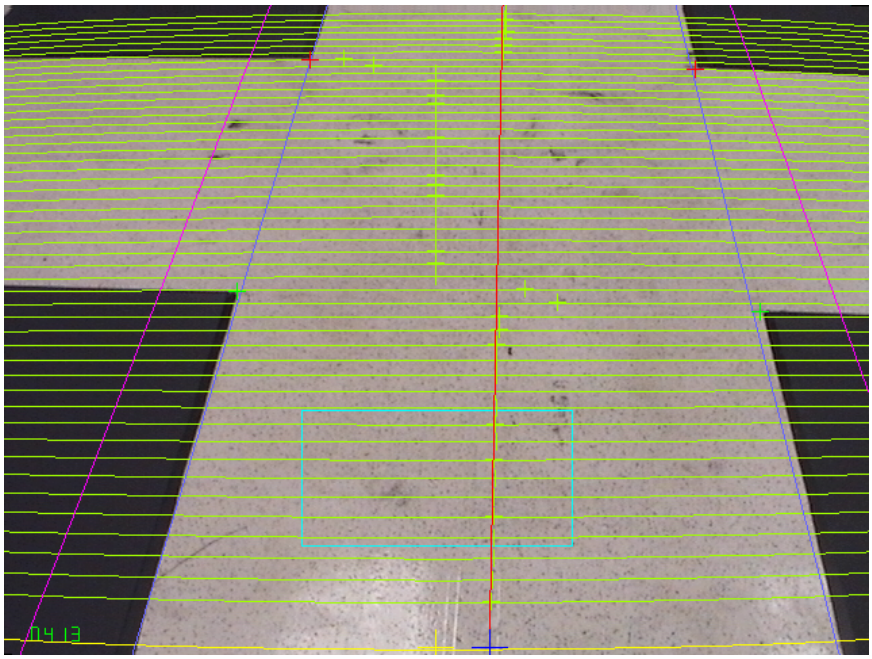


Figure 5-3. Intersection detection image from rotation for flat intersection Run 1. Vehicle positioned 48 inches from intersection and rotated -4° from center of path.

The intersection detection errors for the right side of the path were higher than the left due to the inaccuracies in the camera model. As seen in Figure 5-1, points in the right side of the image showed more error than points in the left.

The average results and a combined count of false negatives and false positives for the three runs are reported in Table 5-17. Both left and right intersection points were averaged for the same distance. Average errors for the beginning of intersections in general increased the farther the intersection was from the vehicle. Average errors for the end of intersections were higher than the beginning of intersections because the end of an intersection is farther away than the beginning.

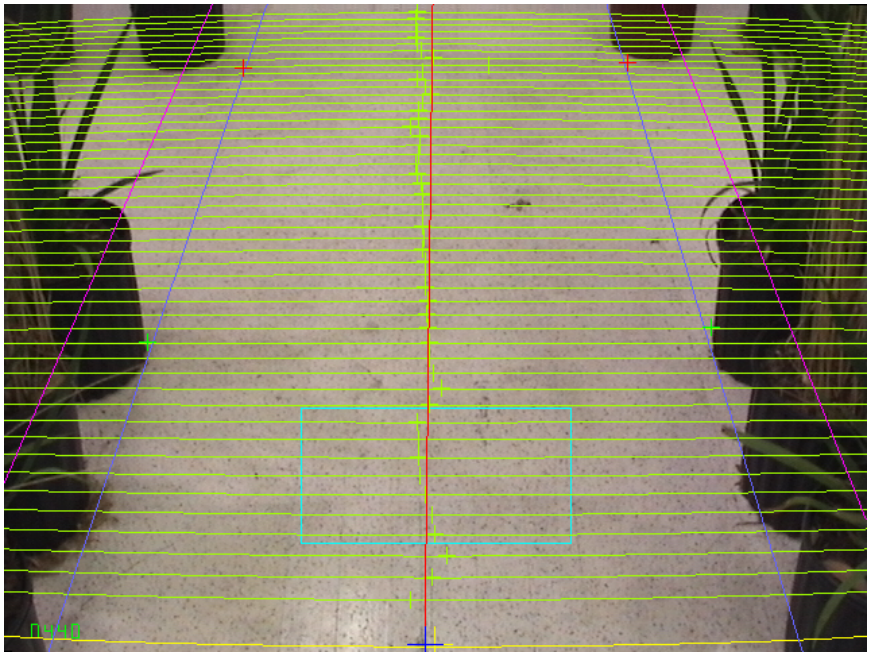


Figure 5-4. Intersection detection image from straight translation for plant intersection Run 1. Vehicle positioned 48 inches from intersection.

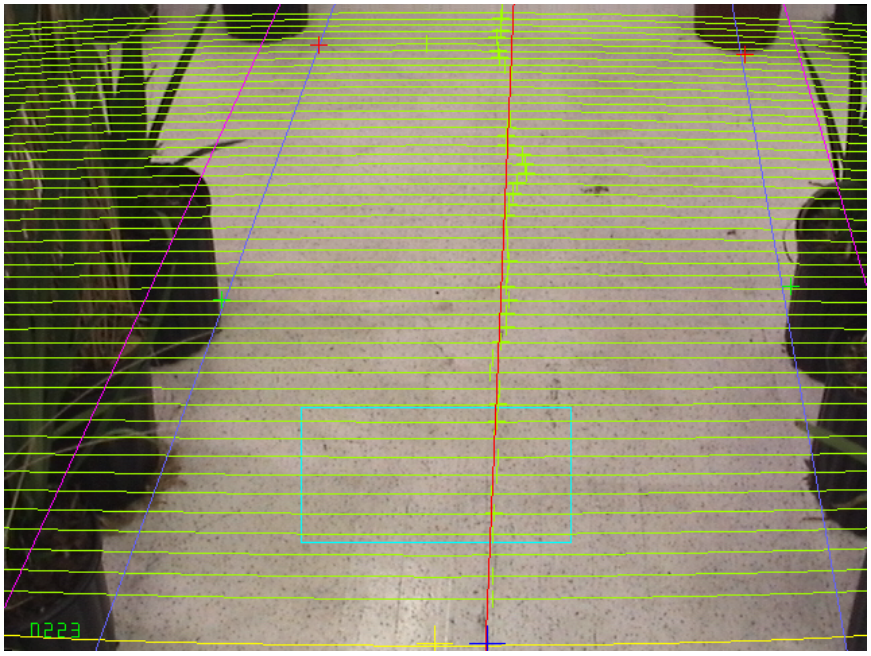


Figure 5-5. Intersection detection image from rotation for plant intersection Run 3. Vehicle positioned 48 inches from intersection and rotated -4° from center of path.

The beginnings of the intersections were not found at the 72-inch distance because the intersection detection algorithm was set to only detect intersection widths greater than 16 inches (the width of the vehicle) during this experiment. The visible width of the

intersecting path is less than this value. Figure 5-7 shows a screen image from this position.

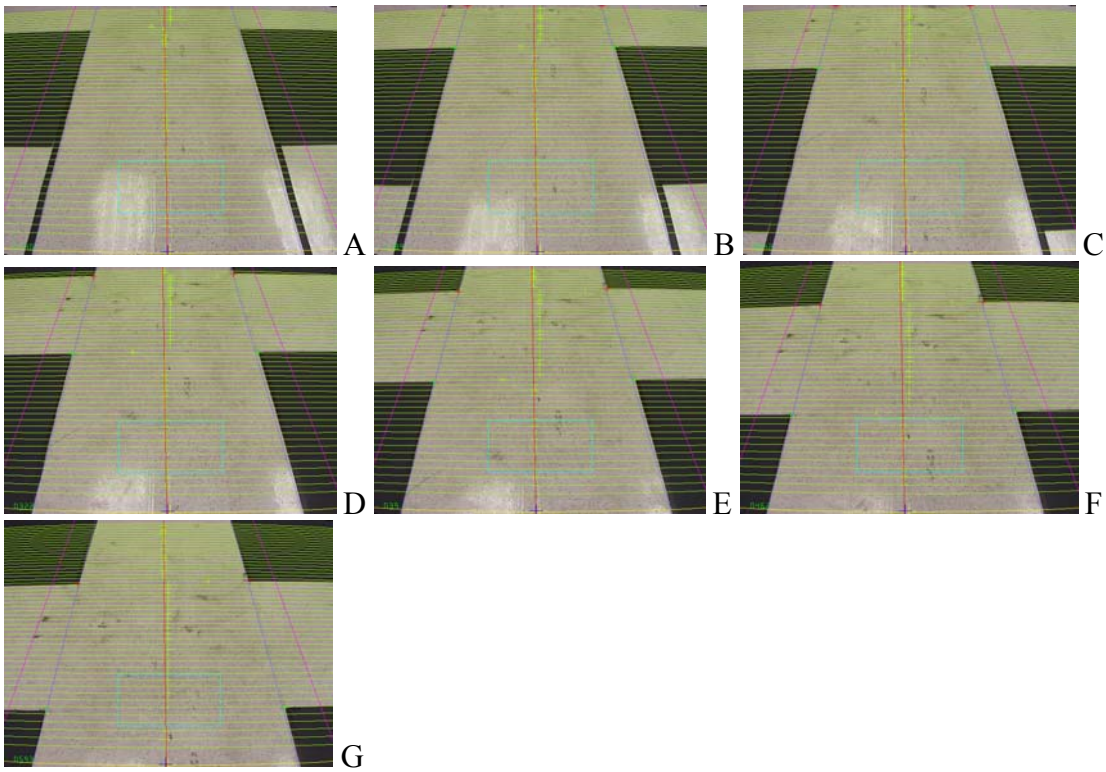


Figure 5-6. Intersection detection images of flat intersection by straight translation Run 1. Distance from beginning of intersection in inches: A) 72, B) 66, C) 60, D) 54, E) 48, F) 42, G) 36.

Table 5-14. Flat intersection, straight translation Run 1.

Distance from Intersection (in)	Left Begin Error (in)	Left End Error (in)	Right Begin Error (in)	Right End Error (in)
72	FN	NV	FN	NV
66	1.02	NV	1.18	NV
60	1.10	NV	1.34	NV
54	0.98	1.27	1.11	1.69
48	0.88	1.17	1.06	1.61
42	0.78	1.16	1.10	1.56
36	0.54	0.81	0.80	1.05

Straight translation: Plant intersection

The same experiment was repeated for the plant intersection. Figure 5-8 shows the intersection detection images taken over a range of distances from Run 1 of the straight translation for plant intersection experiment.

Table 5-15. Flat intersection, straight translation Run 2.

Distance from Intersection (in)	Left Begin Error (in)	Left End Error (in)	Right Begin Error (in)	Right End Error (in)
72	FN	NV	FN	NV
66	0.89	NV	1.05	NV
60	0.82	NV	1.10	NV
54	0.85	0.97	0.97	1.53
48	0.67	0.70	0.93	1.42
42	0.50	0.67	0.84	1.19
36	0.48	0.28	0.79	0.95

Table 5-16. Flat intersection, straight translation Run 3.

Distance from Intersection (in)	Left Begin Error (in)	Left End Error (in)	Right Begin Error (in)	Right End Error (in)
72	FN	NV	FN	NV
66	1.23	NV	1.31	NV
60	1.30	NV	1.39	NV
54	1.31	1.73	1.49	2.14
48	0.93	1.23	1.15	1.68
42	0.78	1.11	1.09	1.53
36	0.57	0.62	0.77	1.00

Table 5-17. Flat intersection, straight translation combined results.

Distance from Intersection (in)	Beginning Average Error (in)	Beginning Total False Negatives	Beginning Total False Positives	End Average Error (in)	End Total False Negatives	End Total False Positives
72	–	6	0	–	0	0
66	1.11	0	0	–	0	0
60	1.17	0	0	–	0	0
54	1.12	0	0	1.56	0	0
48	0.93	0	0	1.30	0	0
42	0.85	0	0	1.20	0	0
36	0.66	0	0	0.79	0	0
Average	0.97	0.86	0.00	1.21	0.00	0.00

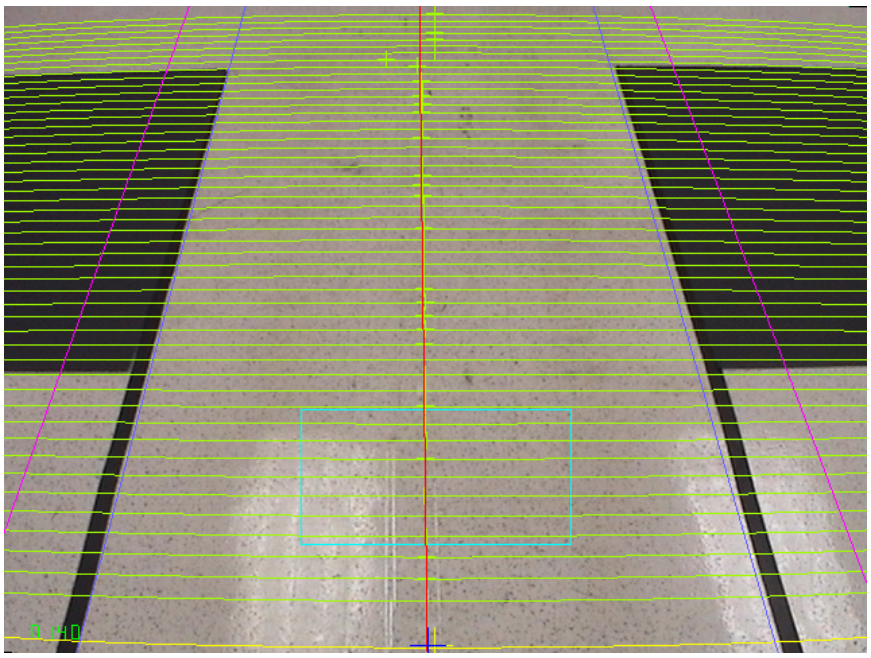


Figure 5-7. Screen image 72 inches from flat intersection.

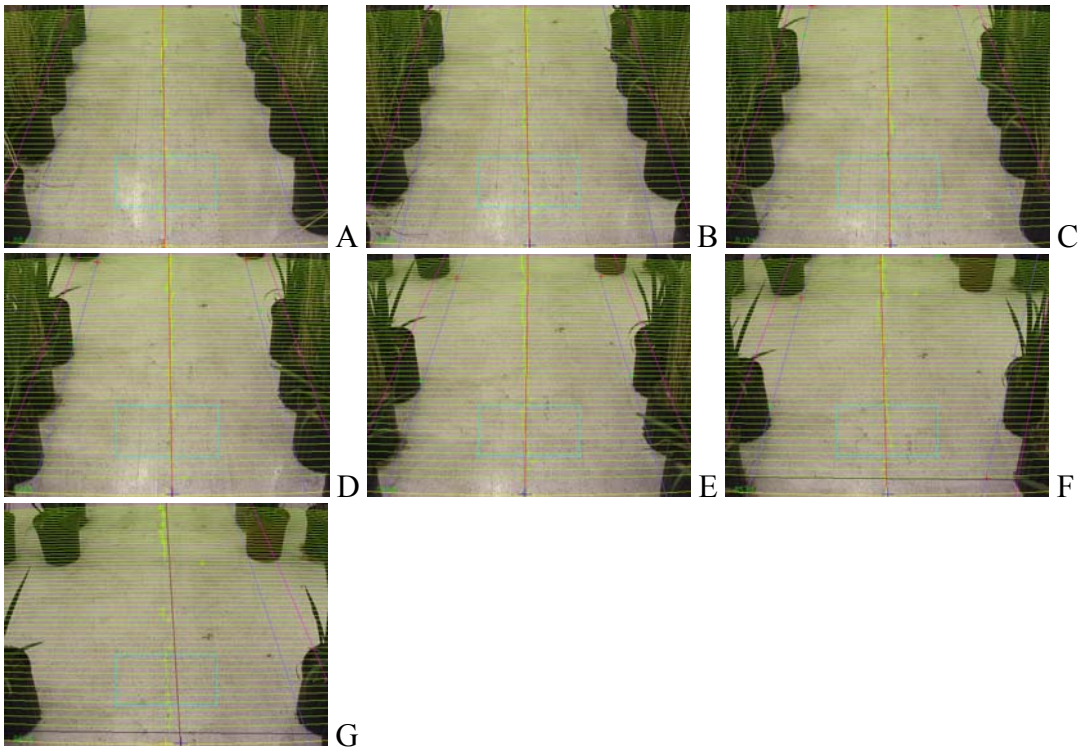


Figure 5-8. Intersection detection images of plant intersection by straight translation Run 1. Distance from beginning of intersection in inches: A) 72, B) 66, C) 60, D) 54, E) 48, F) 42, G) 36.

The results from the three intersection detection runs of the plant intersection by translating the vehicle along the center of the path are reported in Table 5-18, Table 5-19, and Table 5-20. The "Distance from Intersection" is the distance the vehicle origin is from the beginning of the intersection. Errors are reported for how accurate the coordinates of the beginning and end of intersections on the left and right edges of the path were found using the intersection detection algorithm. "FN" stands for false negative (the intersection point was visible in the image but not found). "FP" stands for false positive (an intersection point was detected that does not exist). "NV" stands for not visible (the intersection point was not visible in the image at that particular distance). As with the flat intersection test, the right intersection points showed higher error than the left due to errors in the camera model.

Errors with the plant intersection were expected due to the shape of the potted plants. Due to the round shape of the pots, the intersections were also rounded. As a result, detected intersections did not coincide with the exact intersections between the two paths.

Table 5-18. Plant intersection, straight translation Run 1.

Distance from Intersection (in)	Left Begin Error (in)	Left End Error (in)	Right Begin Error (in)	Right End Error (in)
72	FN	NV	FN	NV
66	FN	NV	FN	NV
60	10.28	NV	2.10	NV
54	0.38	0.68	7.93	2.87
48	2.19	0.50	1.10	1.41
42	1.10	1.22	FP	FP
36	FN	FN	FN	FN

The average results and a combined count of false negatives and false positives for the three runs are reported in Table 5-21. Both left and right intersection points were

averaged for the same distance. Average errors for the beginning of intersections in general increased the farther the intersection was from the vehicle. Average errors for the end of intersections were higher than the beginning of intersections because the end of an intersection is farther away than the beginning.

Table 5-19. Plant intersection, straight translation Run 2.

Distance from Intersection (in)	Left Begin Error (in)	Left End Error (in)	Right Begin Error (in)	Right End Error (in)
72	FN	NV	FN	NV
66	FN	NV	FN	NV
60	10.49	NV	FN	NV
54	1.71	4.73	5.41	NV
48	0.74	2.35	8.12	5.11
42	0.99	2.36	FN	FN
36	0.92	2.03	1.42	3.08

Table 5-20. Plant intersection, straight translation Run 3.

Distance from Intersection (in)	Left Begin Error (in)	Left End Error (in)	Right Begin Error (in)	Right End Error (in)
72	FN	NV	FN	NV
66	FN	NV	FN	NV
60	FN	NV	1.30	NV
54	1.38	4.44	4.85	NV
48	0.80	3.17	FP	FP
42	0.64	2.28	FP	FP
36	0.86	1.79	1.46	4.64

The beginnings of the intersections were not found at the 72- and 66-inch distances due to the increased covering of the intersection path by the potted plants at the corners and because the intersection detection algorithm was set to only detect intersection widths greater than 16 inches during this experiment. Figure 5-9 shows a screen image demonstrating the plant coverage of the intersection from 72 inches from the beginning of the intersection.

Table 5-21. Plant intersection, straight translation combined results.

Distance from Intersection (in)	Beginning Average Error (in)	Beginning Total False Negatives	Beginning Total False Positives	End Average Error (in)	End Total False Negatives	End Total False Positives
72	–	6	0	–	0	0
66	–	6	0	–	0	0
60	6.04	2	0	–	0	0
54	3.61	0	0	3.18	0	0
48	2.59	0	1	2.51	0	1
42	0.91	1	2	1.95	1	2
36	1.16	2	0	2.89	2	0
Average	3.98	2.43	0.43	2.63	0.43	0.43

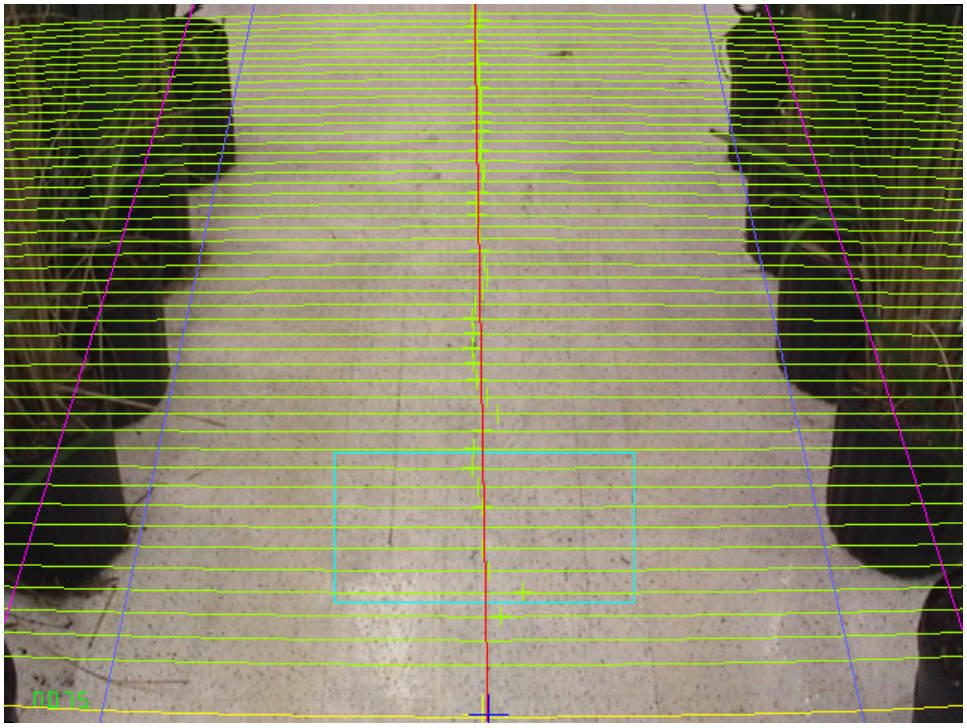


Figure 5-9. Screen image 72 inches from plant intersection.

Another source of error occurred when the best-fit path edge was determined to be along the vertical edge of the plant at the corner of the intersection as show in Figure 5-10. This effect occurred more frequently at closer distances to the intersection and caused the intersection detection algorithm to detect false intersections, as well as miss detecting the real intersection.

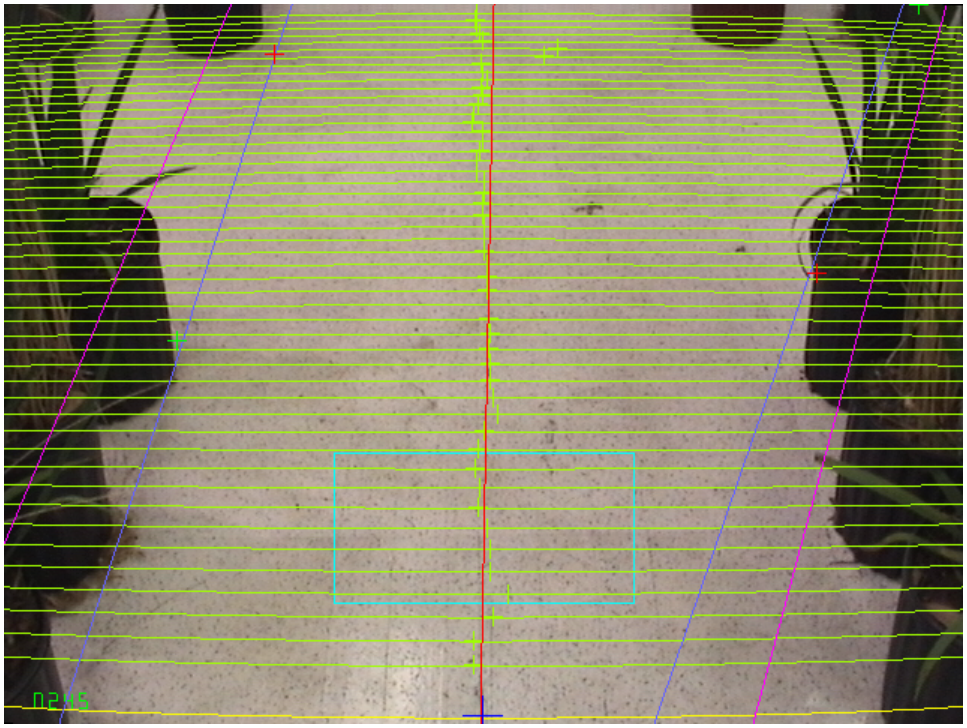


Figure 5-10. Incorrect right path edge at intersection.

Intersection Detection during Rotation

Following are intersection detection results for both the flat intersection and plant intersection setups.

Rotation: Flat intersection

Figure 5-11 shows the intersection detection images taken over a range of angles from Run 1 of the rotation for flat intersection experiment.

The results from the three intersection detection runs of the flat intersection by rotating the vehicle are reported in Table 5-22, Table 5-23, and Table 5-24.

The average results and a combined count of false negatives and false positive for the three runs are reported in Table 5-25. Both left and right intersection points were averaged for the vehicle rotation. Average errors showed higher accuracy of intersection detection when the vehicle was lined up with the path center (0°). Average errors were highest at the greatest angles of rotation (8° and -8°).

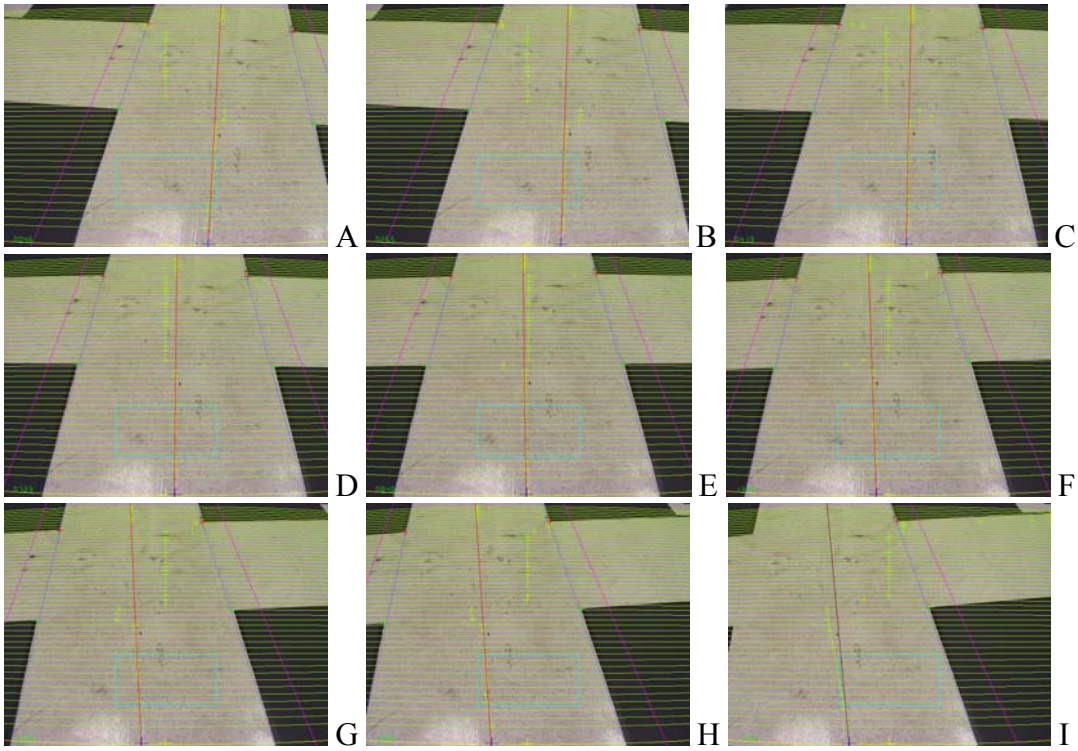


Figure 5-11. Intersection detection images of flat intersection by rotation Run 1. Rotation about vehicle z-axis in degrees: A) -8, B) -6, C) -4, D) -2, E) 0, F) 2, G) 4, H) 6, I) 8.

Table 5-22. Flat intersection, rotation Run 1.

Vehicle Rotation (deg)	Left Begin Error (in)	Left End Error (in)	Right Begin Error (in)	Right End Error (in)
-8	1.23	1.71	1.68	2.15
-6	1.19	1.61	1.25	2.14
-4	1.18	1.46	1.30	1.92
-2	1.09	1.33	1.20	1.92
0	0.97	1.20	1.23	1.90
2	0.85	1.06	1.17	2.70
4	0.89	1.00	1.07	1.79
6	0.90	1.19	1.07	1.74
8	FN	FN	1.13	1.70

As with the straight intersection tests, the right intersection points showed higher error than the left due to errors in the camera model. Higher errors in the right side of the image also contributed to higher errors when the vehicle was angled at -8°. When the

vehicle was angled at -8°, the intersections lied closer to the right side of the image, where the most camera model error occurred.

Table 5-23. Flat intersection, rotation Run 2.

Vehicle Rotation (deg)	Left Begin Error (in)	Left End Error (in)	Right Begin Error (in)	Right End Error (in)
-8	1.11	1.53	2.89	2.20
-6	1.15	1.46	1.25	2.08
-4	1.18	1.46	1.29	2.16
-2	1.12	1.39	1.27	2.07
0	1.09	1.26	1.23	1.77
2	0.98	1.40	1.24	1.83
4	0.91	1.22	1.22	1.93
6	1.19	1.12	1.13	1.70
8	FN	FN	1.23	1.99

Table 5-24. Flat intersection, rotation Run 3.

Vehicle Rotation (deg)	Left Begin Error (in)	Left End Error (in)	Right Begin Error (in)	Right End Error (in)
-8	1.23	1.69	2.89	2.56
-6	1.28	1.68	1.39	2.44
-4	1.24	1.70	1.40	2.36
-2	1.14	1.60	1.40	2.21
0	1.00	1.25	1.23	1.96
2	1.07	1.47	1.38	2.22
4	0.94	1.29	1.26	2.00
6	1.59	1.34	1.29	2.10
8	FN	FN	1.18	1.87

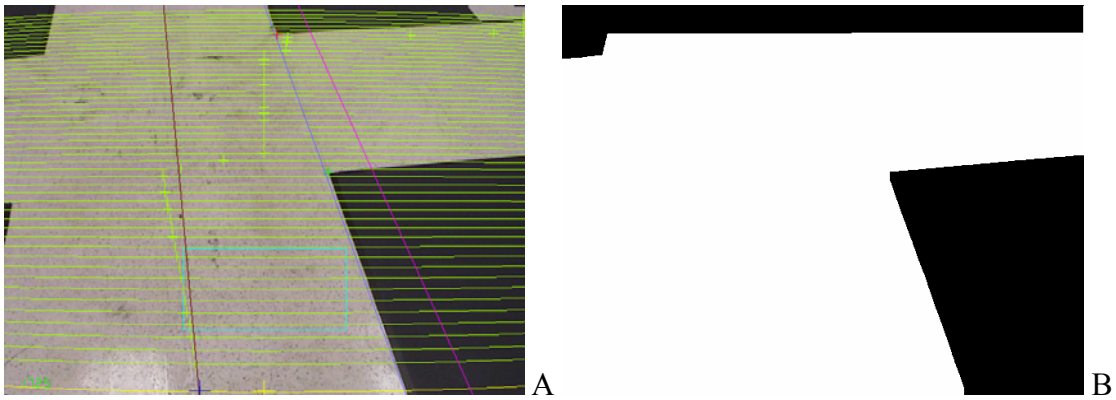
The left intersection was not detected when the vehicle was oriented at 8°. Figure 5-12 shows a screen image and segmented path image from this position. The left path edge was removed during image processing section (described in Chapter 3) due to the small amount of left path edge visible. Without a left path edge, no intersection on the left side can be detected.

Rotation: Plant intersection

The same experiment was repeated for the plant intersection. Figure 5-13 shows the intersection detection images taken over a range of angles from Run 1 of the rotation for plant intersection experiment.

Table 5-25. Flat intersection, rotation combined results.

Vehicle Rotation (deg)	Beginning Average Error (in)	Beginning Total False Negatives	Beginning Total False Positives	End Average Error (in)	End Total False Negatives	End Total False Positives
-8	1.84	0	0	1.98	0	0
-6	1.25	0	0	1.90	0	0
-4	1.27	0	0	1.84	0	0
-2	1.20	0	0	1.75	0	0
0	1.13	0	0	1.56	0	0
2	1.11	0	0	1.78	0	0
4	1.05	0	0	1.54	0	0
6	1.19	0	0	1.53	0	0
8	1.18	3	0	1.86	3	0
Average	1.25	0.33	0.00	1.75	0.33	0.00



A B

Figure 5-12. Images from vehicle orientation of 8°. A) Screen image. B) Segmented path image.

The results from the three intersection detection runs of the plant intersection by rotating the vehicle are reported in Table 5-26, Table 5-27, and Table 5-28.

The average results and a combined count of false negatives and false positive for the three runs are reported in Table 5-29. Both left and right intersection points were averaged for the vehicle rotation. Average errors showed higher accuracy of intersection

detection when the vehicle was lined up with the path center (0°). Average errors were highest at the greatest angles or rotation (8° and -8°).

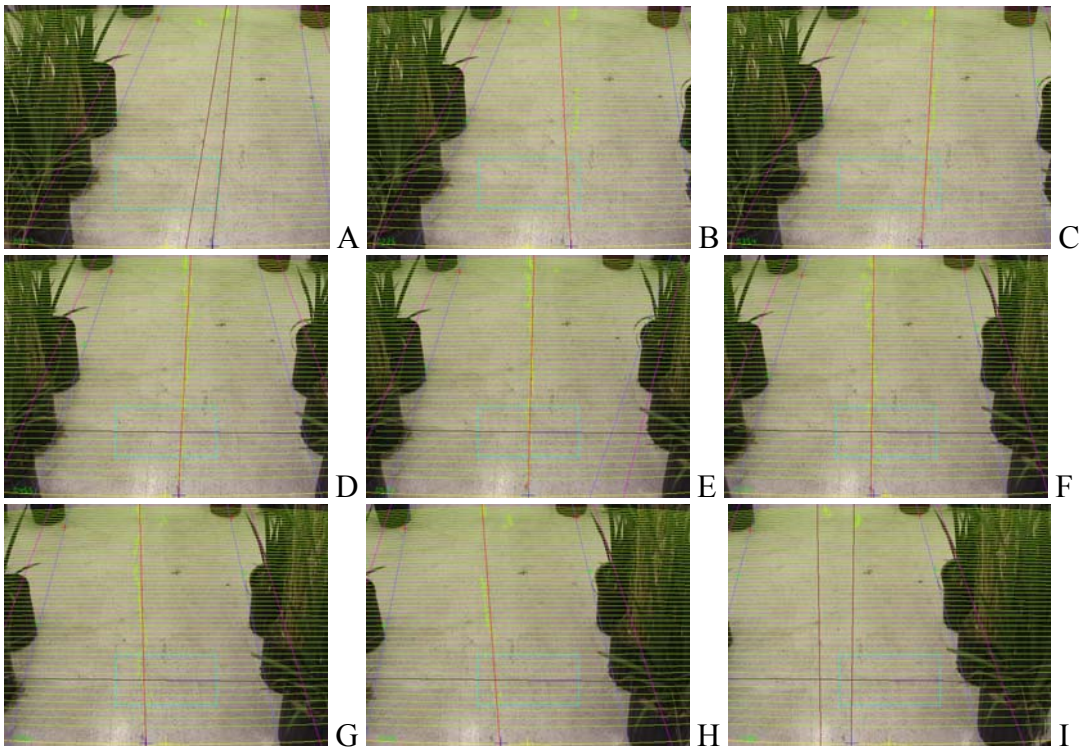


Figure 5-13. Intersection detection images of plant intersection by rotation Run 1. Rotation about vehicle z-axis in degrees: A) -8, B) -6, C) -4, D) -2, E) 0, F) 2, G) 4, H) 6, I) 8.

Table 5-26. Plant intersection, rotation Run 1.

Vehicle Rotation (deg)	Left Begin Error (in)	Left End Error (in)	Right Begin Error (in)	Right End Error (in)
-8	1.04	3.17	5.27	5.04
-6	0.82	3.16	2.57	5.75
-4	0.66	4.23	11.52	10.64
-2	5.24	3.26	FN	FN
0	0.92	3.46	FN	FN
2	0.58	2.54	9.91	5.83
4	0.52	2.25	9.94	5.95
6	0.37	3.18	9.69	5.81
8	12.71	3.34	10.20	5.72

Like the errors associated with the straight translation test, the rotation test saw errors due to the slight miscalculation of path edge lines, as well as errors due to the rounded intersections created by the round pots. Figure 5-14 demonstrates errors in beginning and end of intersection points due to a slightly miscalculated path edge. This occurred frequently on the right path edge and is reflected in the above tables. The cause of the path edge miscalculation was the slight misalignment of the last plant on the right path edge before the intersection, which contributed to the high errors in the detected beginnings and endings of that intersection during testing.

Table 5-27. Plant intersection, rotation Run 2.

Vehicle Rotation (deg)	Left Begin Error (in)	Left End Error (in)	Right Begin Error (in)	Right End Error (in)
-8	FN	FN	7.86	5.10
-6	1.27	3.30	14.92	15.14
-4	2.19	2.66	FN	FN
-2	1.05	3.91	FN	FN
0	0.58	3.51	FN	FN
2	0.81	2.82	9.46	5.54
4	1.25	3.45	10.01	5.50
6	0.68	2.77	10.13	5.54
8	8.88	3.41	10.54	6.10

Table 5-28. Plant intersection, rotation Run 3.

Vehicle Rotation (deg)	Left Begin Error (in)	Left End Error (in)	Right Begin Error (in)	Right End Error (in)
-8	FP	FP	8.00	5.13
-6	FN	FN	2.22	5.07
-4	1.23	3.15	3.36	5.09
-2	1.19	2.76	FP	FP
0	0.97	3.17	FN	FN
2	1.08	2.29	9.82	5.25
4	1.06	2.83	9.84	5.02
6	0.73	3.51	10.10	5.40
8	7.94	7.86	FN	FN

Table 5-29. Plant intersection, rotation combined results.

Vehicle Rotation (deg)	Beginning Average Error (in)	Beginning Total False Negatives	Beginning Total False Positives	End Average Error (in)	End Total False Negatives	End Total False Positives
-8	5.54	1	1	4.61	1	1
-6	4.36	1	0	6.49	1	0
-4	3.79	1	0	5.15	1	0
-2	2.49	2	1	3.31	2	1
0	0.83	3	0	3.38	3	0
2	5.28	0	0	4.04	0	0
4	5.44	0	0	4.17	0	0
6	5.28	0	0	4.37	0	0
8	10.05	1	0	5.29	1	0
Average	4.78	1.00	0.22	6.58	1.00	0.22

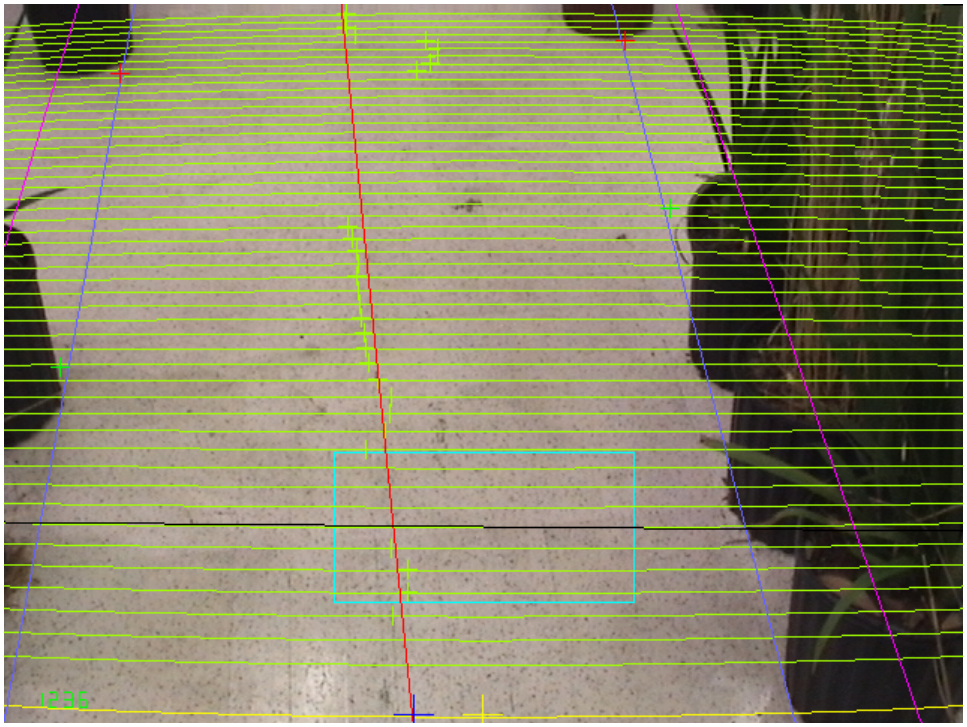
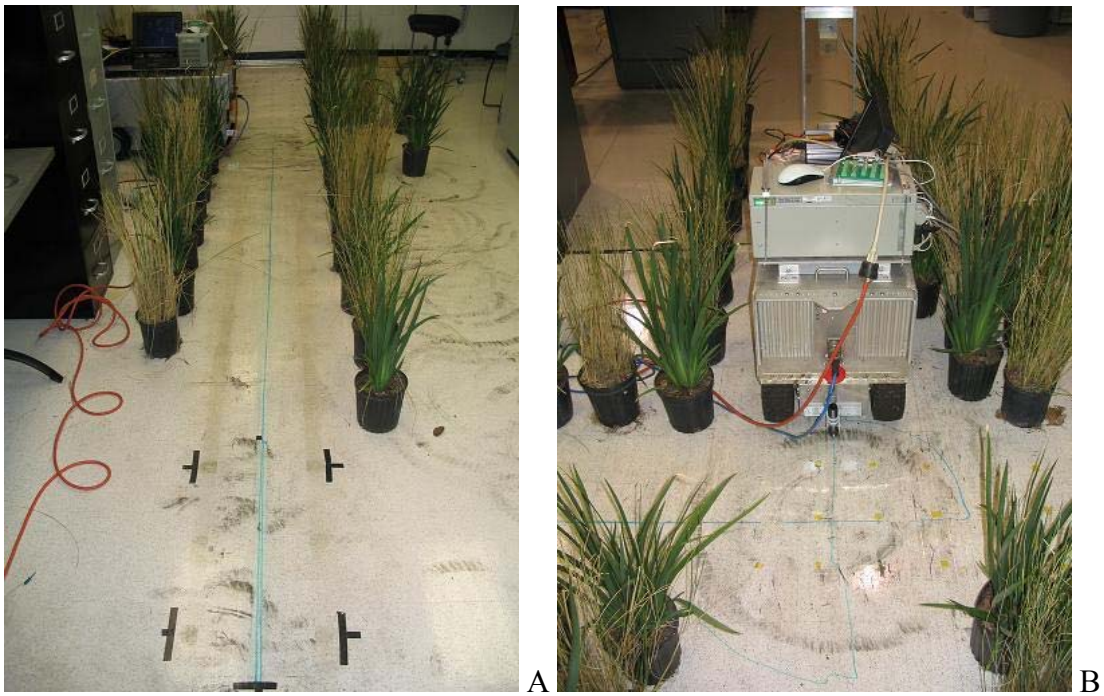


Figure 5-14. Intersection detection error from miscalculated path edge. See Table 5-26, Run 2, 6° rotation, for error measurements.

Intersection Navigation

Results were obtained from experiments on a flat intersection lined with tape and plant intersection lined with potted plants as described in the previous chapter. For each type of intersection, the vehicle was instructed to make a left turn at the intersection and

drive straight through the intersection. Measurements were made from path lines left by markers mounted onto the front and back of the vehicle. Figure 5-15 shows the path marks left by the vehicle going into and out of a left turn during an experimental run.



A B

Figure 5-15. Path marking for left turn experiment. A) View from first path. B) View from second path.

Turning Navigation Experiments

The following are intersection navigation results for both the flat intersection and plant intersection setups. The algorithm ran at an average of 5 Hz while following the first path, 10 Hz while navigating through the intersection with the visual odometer, 8 Hz while turning, and 5 Hz while following the second path. The difference in run speed depended on the program code run during that process. Path following ran slowest at 5 Hz due to the image processing, path analysis, and path following control algorithms. Turning ran slightly faster than path following because the path following control algorithms did not need to be executed. Traveling through the intersection ran the fastest because no image processing or path analysis had to be carried out.

Turning navigation: Flat intersection

Figure 5-16 shows the resulting paths for the vehicle completing a left turn with the flat intersection setup. Two separate lines were drawn for the front and back paths. Table 5-30 reports the path error along the center of the first path before going into the turn for the front of the vehicle. Table 5-31 reports the path error along the center of the second path after turning for the front of the vehicle. Table 5-32 and Table 5-33 report the path error along the center of the first and second paths for the back of the vehicle. Average error along the first path was under 0.5 inches. Average error for the second path was slightly over an inch, due to the initial displacement going into the second path after turning.

The turning errors at the intersection for each of the three runs are plotted in Figure 5-17, Figure 5-18, and Figure 5-19. F_{Start} and F_{Finish} represent the location of the front of the vehicle at the start and end of the turn. B_{Start} and B_{Finish} represent the location of the back of the vehicle at the start and end of the turn. TC_{Start} and TC_{Finish} represent the location of the vehicle's turning center at the start and end of the turn. The turning center was calculated by finding the midpoint between the front and back of the vehicle. Note that the turning center drifts during the turn, which contributed to vehicle position error relative to the second path after turning. This drift was due to the mechanical properties of the physical vehicle, not the navigation algorithm. Table 5-34 shows the calculated turning errors at the intersection for both the vehicle position and orientation.

The average vehicle orientation going into the turn was –0.72° off the first path center, showing accurate vehicle control using the visual odometer. The average vehicle orientation going out of the turn was –2.86° off the second path center. This demonstrated a combination of accurate vehicle position in the intersection going into the

turn, detection of the next path, and stopping of the turn when the center of the next path was in line with the vehicle center.

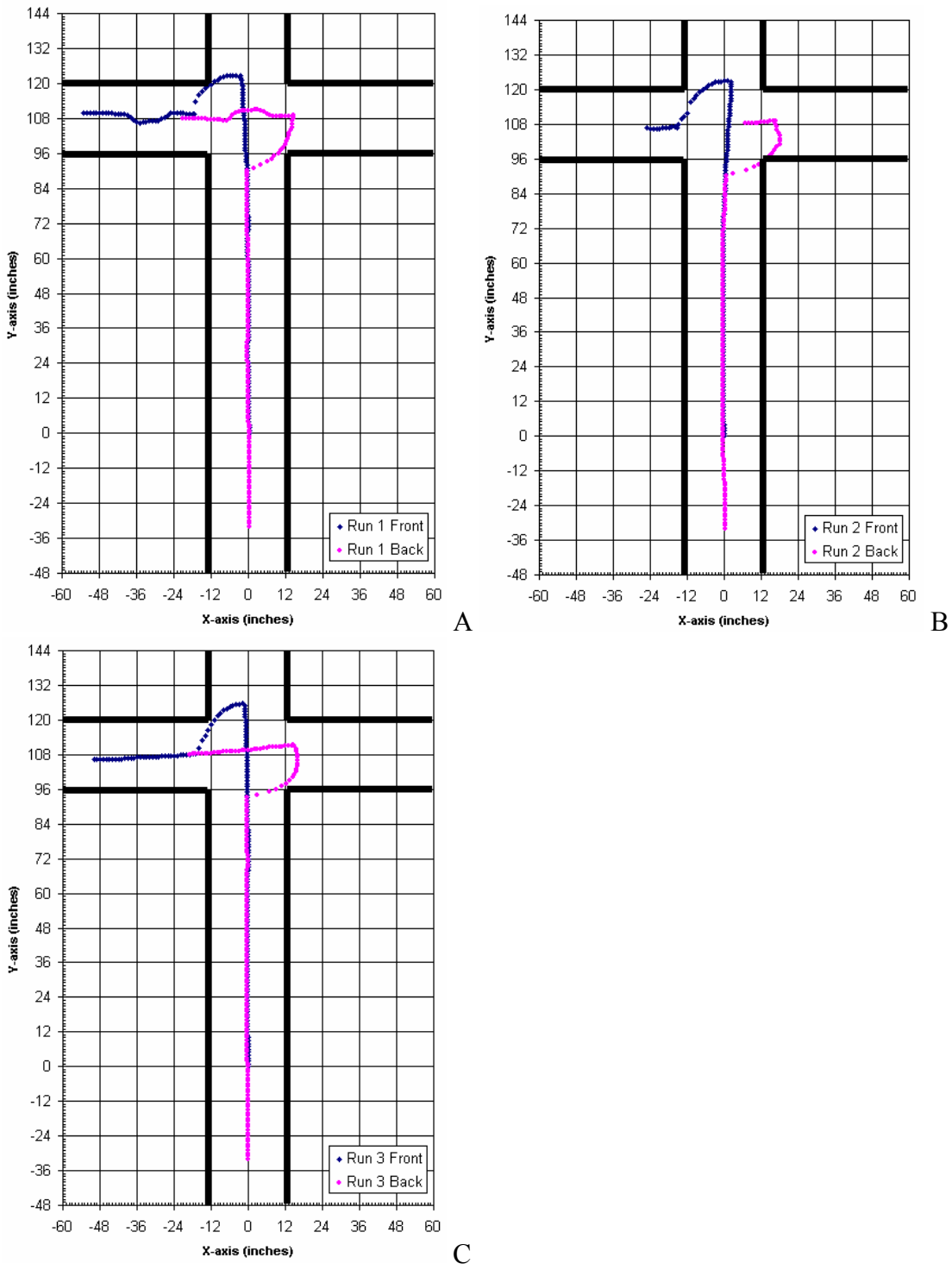


Figure 5-16. Recorded vehicle paths for flat intersection turning experiment. A) Run 1. B) Run 2. C) Run 3.

Table 5-30. Flat intersection, front path error before turn.

Run	Average Error (in)	RMS Error (in)	Max Error (in)	Standard Deviation (in)
1	0.46	0.80	2.50	0.65
2	0.60	0.82	2.25	0.57
3	0.30	0.39	1.00	0.25
Average	0.45	0.67	1.92	0.49

Table 5-31. Flat intersection, front path error after turn.

Run	Average Error (in)	RMS Error (in)	Max Error (in)	Standard Deviation (in)
1	1.36	1.44	1.75	0.48
2	1.18	1.25	1.50	0.43
3	1.24	1.75	6.50	1.25
Average	1.26	1.48	3.25	0.72

Table 5-32. Flat intersection, back path error before turn.

Run	Average Error (in)	RMS Error (in)	Max Error (in)	Standard Deviation (in)
1	0.15	0.22	0.50	0.16
2	0.33	0.38	0.75	0.18
3	0.17	0.21	0.25	0.12
Average	0.22	0.27	0.50	0.15

Table 5-33. Flat intersection, back path error after turn.

Run	Average Error (in)	RMS Error (in)	Max Error (in)	Standard Deviation (in)
1	1.11	1.46	3.00	0.96
2	0.68	0.72	1.00	0.25
3	1.58	1.86	3.50	0.99
Average	1.13	1.35	2.50	0.73

Turning navigation: Plant intersection

Figure 5-20 shows the resulting paths for the vehicle completing a left turn with the plant intersection setup. Two separate lines were drawn for the front and back paths. Table 5-35 reports the path error along the center of the first path before going into the

turn for the front of the vehicle. Table 5-36 reports the path error along the center of the second path after turning for the front of the vehicle. Table 5-37 and Table 5-38 report the path error along the center of the first and second paths for the back of the vehicle. Average error along the first path was under 0.5 inches. Average error for the second path was 3.08 inches, due to the initial displacement going into the second path after turning.

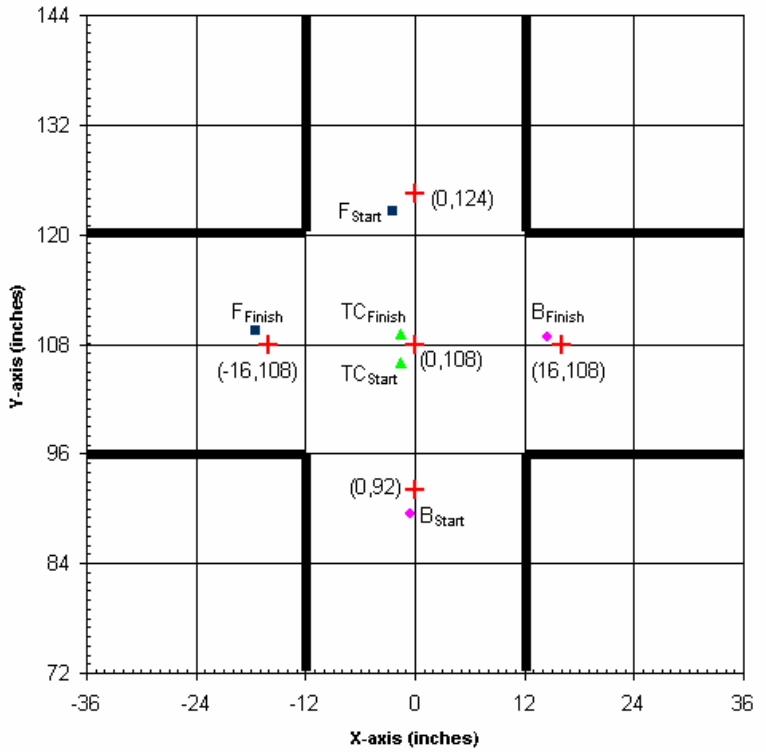


Figure 5-17. Flat intersection, turning errors for Run 1.

The turning errors at the intersection for each of the three runs are plotted in Figure 5-21, Figure 5-22, and Figure 5-23. F_{Start} and F_{Finish} represent the location of the front of the vehicle at the start and end of the turn. B_{Start} and B_{Finish} represent the location of the back of the vehicle at the start and end of the turn. TC_{Start} and TC_{Finish} represent the location of the vehicle's turning center at the start and end of the turn. The turning center was calculated by finding the midpoint between the front and back of the vehicle. Table

5-39 shows the calculated turning errors at the intersection for both the vehicle position and orientation.

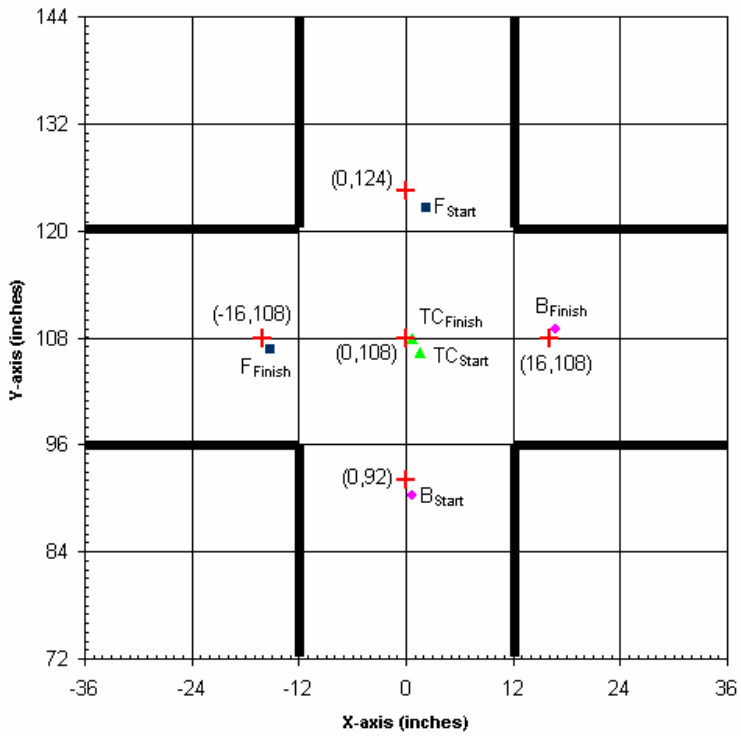


Figure 5-18. Flat intersection, turning errors for Run 2.

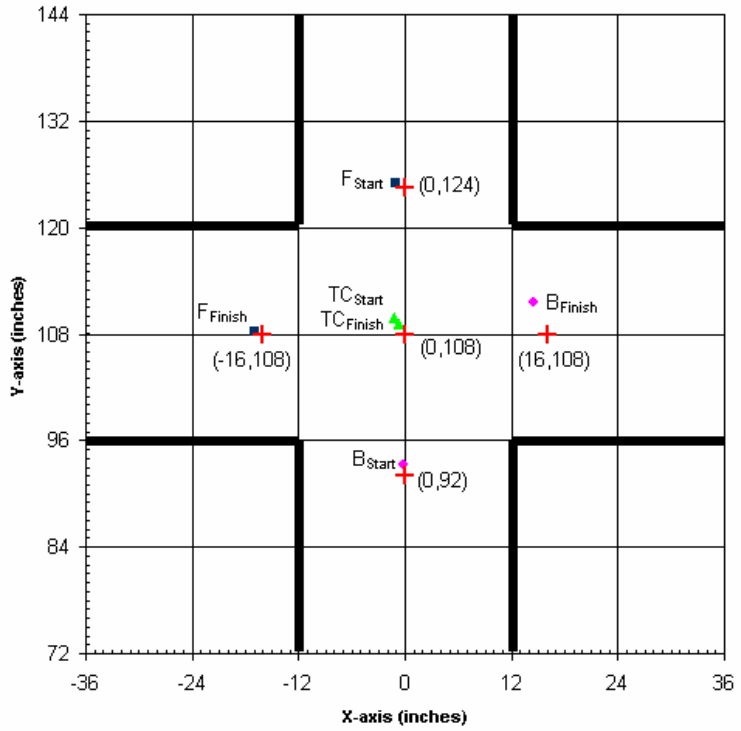


Figure 5-19. Flat intersection, turning errors for Run 3.

Table 5-34. Flat intersection, turning errors.

Run	Begin Turn Error-Front (in)	Begin Turn Error-Back (in)	End Turn Error-Front (in)	End Turn Error-Back (in)	Begin Turn Orientation (deg)	End Turn Orientation (deg)	Turning Center Start Error (in)	Turning Center End Error (in)
1	2.92	2.55	2.12	1.68	-3.47	1.34	2.50	1.88
2	2.70	1.90	1.46	1.25	2.66	-4.02	2.21	0.76
3	1.41	1.27	1.03	3.81	-1.35	-5.89	1.29	2.25
Average	2.34	1.91	1.54	2.24	-0.72	-2.86	2.00	1.63
Min	1.41	1.27	1.03	1.25	-3.47	-5.89	1.29	0.76
Max	2.92	2.55	2.12	3.81	2.66	1.34	2.50	2.25

The average vehicle orientation going into the turn was 1.2° off the first path center, showing accurate vehicle control using the visual odometer. The average vehicle orientation going out of the turn was –4.44° off the second path center. Despite the ability for all three runs to complete the turn with satisfactory vehicle orientation going into and out of the turn, turning error did play a significant role in the path errors seen in the second path after the turn. Several factors contributed to the turning errors: 1) inaccurate calculation of the turning point, 2) visual odometer error when driving the vehicle to the turning point, and 3) drifting of the vehicle during the turn.

Figure 5-21 and Figure 5-22 show that the vehicle began turning too far down the first path for runs 1 and 2. Part of this was due to inaccurate calculation of the turning point, which is the point the vehicle needs to drive to before beginning its turns (see "Turning Navigation: Step 3" in Chapter 3 for definition). Figure 5-24 shows the turning point calculated for Run 2. It is further down the path than desired because the detected beginning of the intersection was slightly off. Figure 5-25 shows the turning point calculated for Run 3. During this run, the turning point was calculated more accurately due to better intersection detection results.

Visual odometer errors were due to the camera model and general error, as seen in the visual odometer experiments. The physical properties of the vehicle contributed to the drifting error.

Note that both runs saw problems detecting the intersection on the other side of the path due to inaccurate calculations of the best-fit path edge line as mentioned in the previous intersection detection results for the plant intersection setup.

Straight Navigation Experiments

The following are intersection navigation results for both the flat intersection and plant intersection setups. The intersection detection algorithm ran at an average of 5 Hz while following the first path, 4 Hz while visually driving through the intersection, and 5 Hz while following the second path. The difference in run speed depended on the program code run during that process. Path following ran slowest at 5 Hz due to the image processing, path analysis, and path following control algorithms. Traveling through the intersection ran the slowest because the visual odometer was run in addition to the path following algorithms.

Straight navigation: Plant intersection

Figure 5-27 shows the resulting paths for the vehicle driving straight through the plant intersection setup. Two separate lines were drawn for the front and back paths. Table 5-42 reports the path error along the center of the path for the front of the vehicle. Table 5-43 reports the path error along the center of the path for the back of the vehicle. Average error along the path was under 0.50 inches.

Table 5-35. Plant intersection, front path error before turn.

Run	Average Error (in)	RMS Error (in)	Max Error (in)	Standard Deviation (in)
1	0.33	0.43	1.00	0.28
2	0.59	0.91	2.25	0.70
3	0.45	0.51	0.75	0.23
Average	0.46	0.62	1.33	0.41

Table 5-36. Plant intersection, front path error after turn.

Run	Average Error (in)	RMS Error (in)	Max Error (in)	Standard Deviation (in)
1	2.79	3.06	5.00	1.27
2	4.83	4.87	5.75	0.59
3	1.61	1.67	2.25	0.45
Average	3.08	3.20	4.33	0.77

Table 5-37. Plant intersection, back path error before turn.

Run	Average Error (in)	RMS Error (in)	Max Error (in)	Standard Deviation (in)
1	0.15	0.23	0.50	0.18
2	0.16	0.34	1.00	0.31
3	0.21	0.29	0.50	0.19
Average	0.17	0.29	0.67	0.23

Table 5-38. Plant intersection, back path error after turn.

Run	Average Error (in)	RMS Error (in)	Max Error (in)	Standard Deviation (in)
1	4.71	5.04	8.00	1.82
2	7.21	7.33	10.00	1.35
3	2.21	2.31	3.00	0.69
Average	4.71	4.89	7.00	1.29

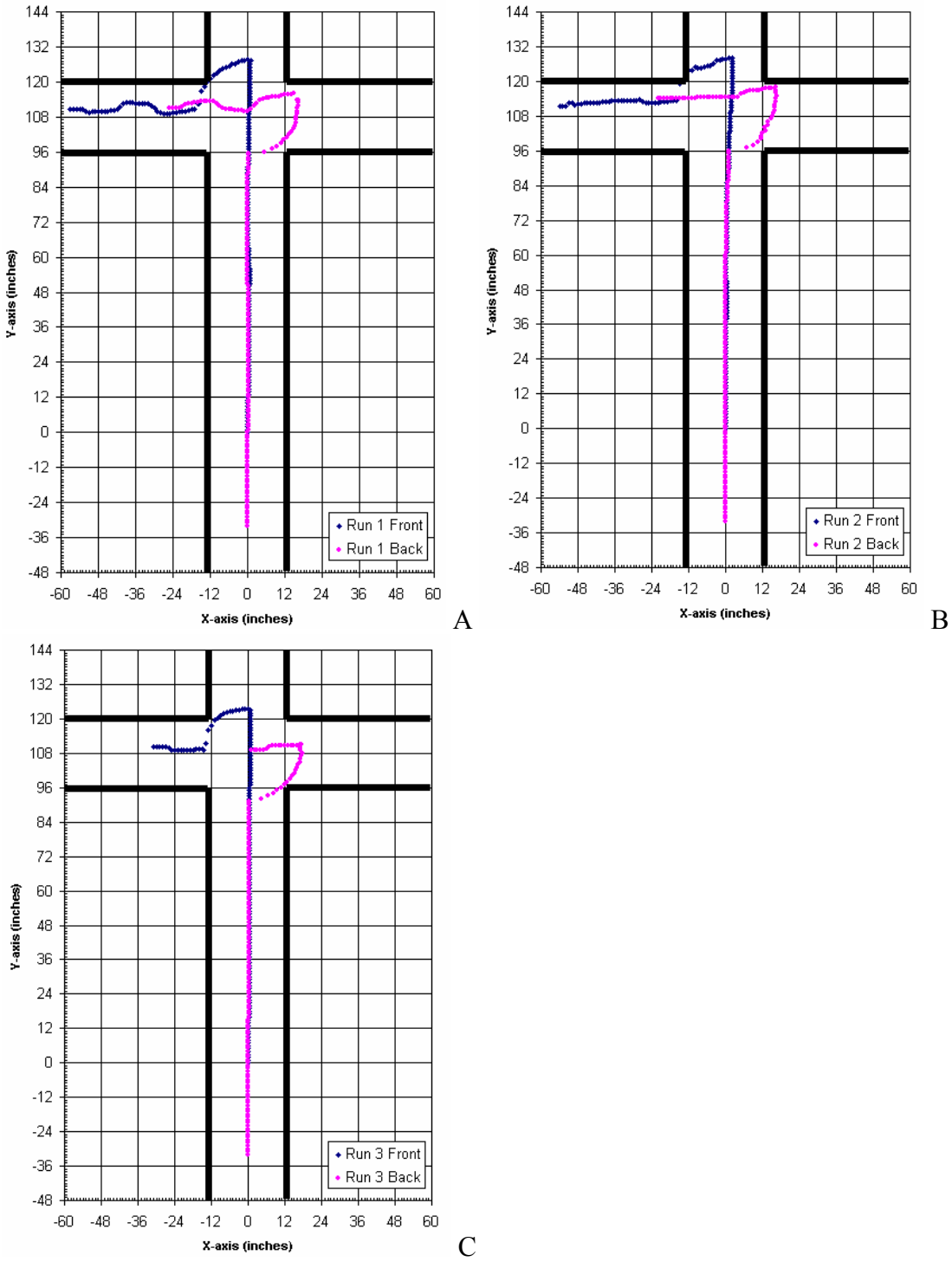


Figure 5-20. Recorded vehicle paths for plant intersection turning navigation experiment. A) Run 1. B) Run 2. C) Run 3.

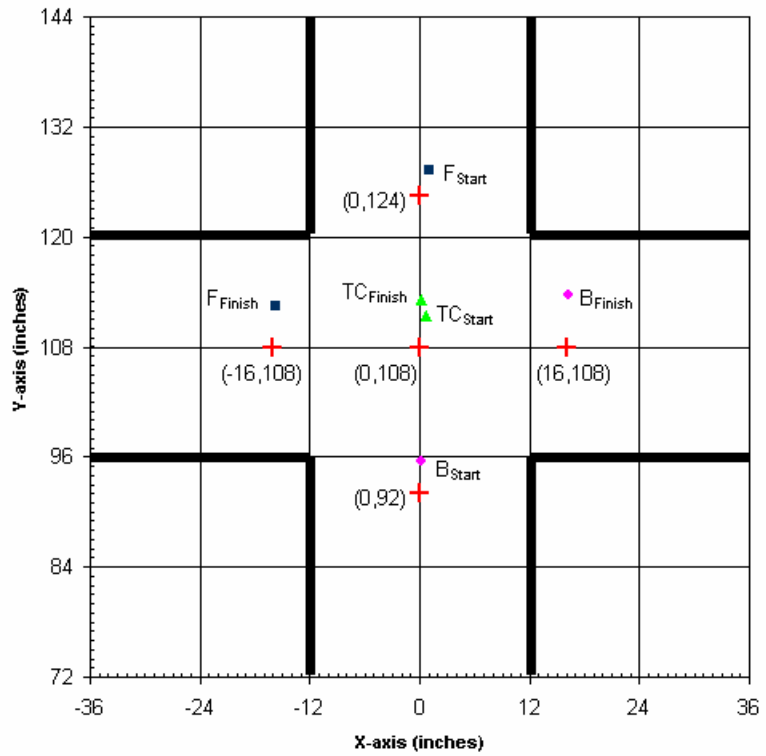


Figure 5-21. Plant intersection, turning errors for Run 1.

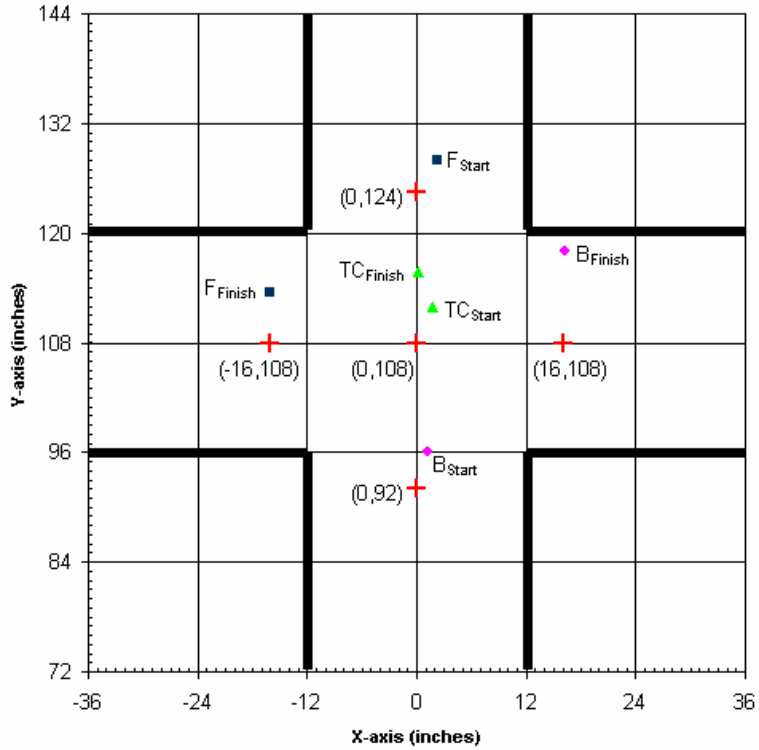


Figure 5-22. Plant intersection, turning errors for Run 2.

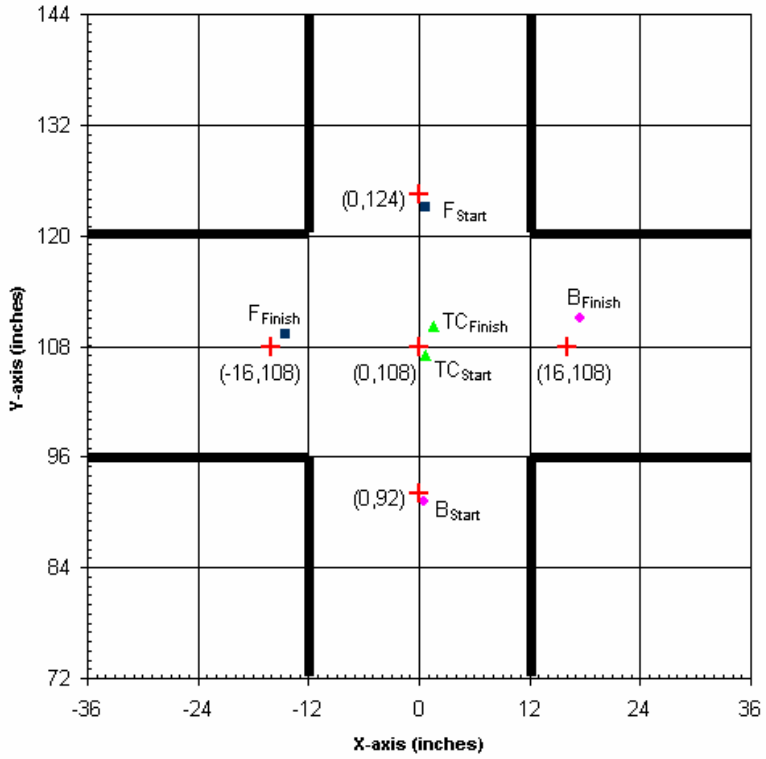


Figure 5-23. Plant intersection, turning errors for Run 3.

Table 5-39. Plant intersection, turning errors.

Run	Begin Turn Error-Front (in)	Begin Turn Error-Back (in)	End Turn Error-Front (in)	End Turn Error-Back (in)	Begin Turn Orientation (deg)	End Turn Orientation (deg)	Turning Center Start Error (in)	Turning Center End Error (in)
1	3.40	3.51	4.51	5.76	1.35	-2.24	3.43	5.13
2	4.59	4.19	5.50	10.00	1.79	-7.94	4.37	7.75
3	1.25	0.90	1.95	3.35	0.45	-3.13	1.08	2.60
Average	3.08	2.87	3.99	6.37	1.20	-4.44	2.96	5.16
Min	1.25	0.90	1.95	3.35	0.45	-7.94	1.08	2.60
Max	4.59	4.19	5.50	10.00	1.79	-2.24	4.37	7.75

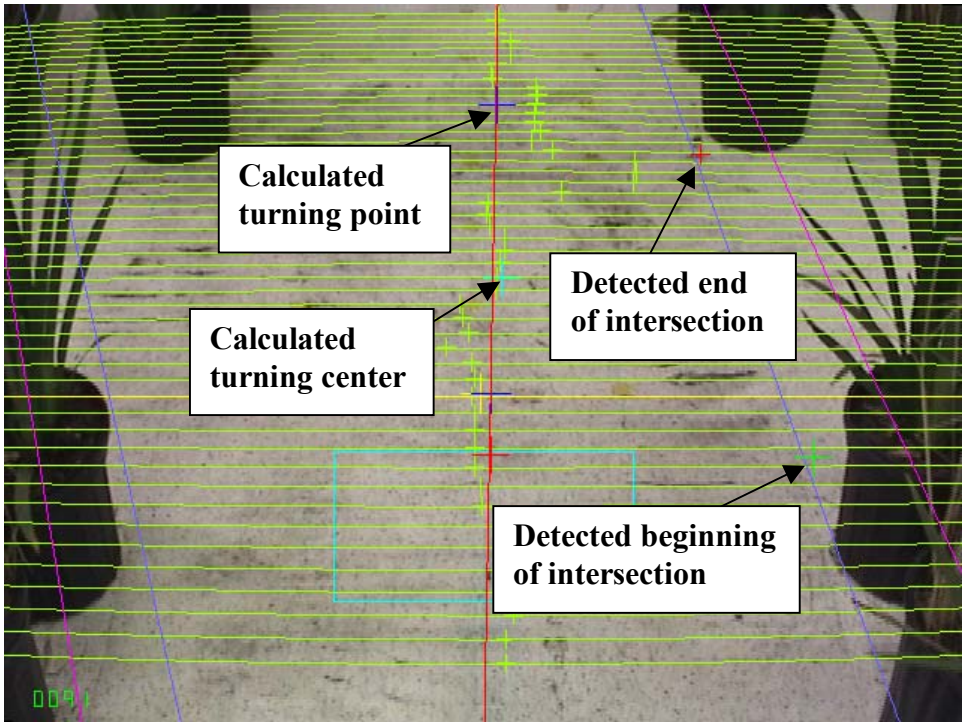


Figure 5-24. Intersection detection for plant intersection navigation. Turning navigation experiment Run 2.

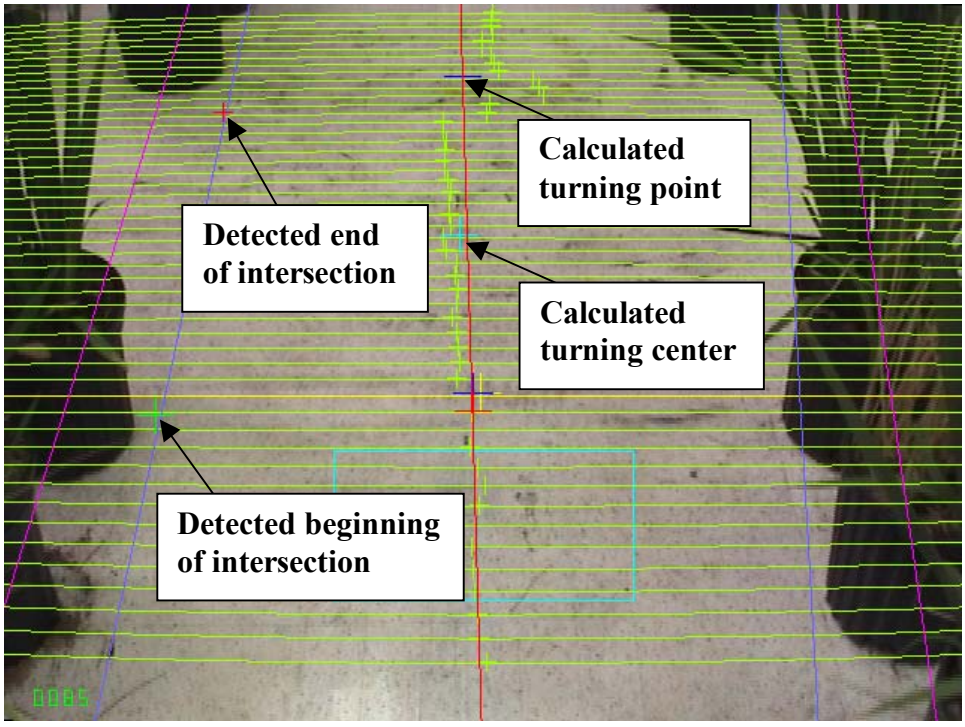
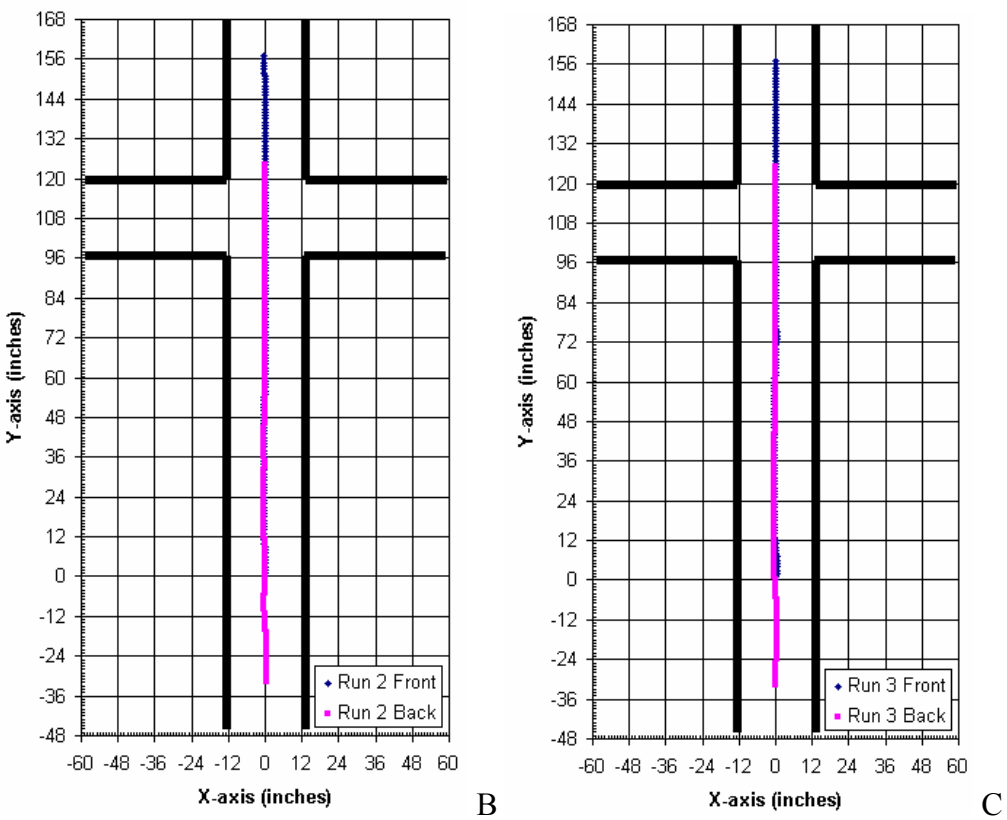


Figure 5-25. Intersection detection for plant intersection navigation. Turning navigation experiment Run 3.



A B C

Figure 5-26. Recorded vehicle paths for flat intersection straight navigation experiment. A) Run 1. B) Run 2. C) Run 3.

Table 5-40. Flat intersection, front path error for straight navigation experiment.

Run	Average Error (in)	RMS Error (in)	Max Error (in)	Standard Deviation (in)
1	0.20	0.27	0.50	0.19
2	0.08	0.14	0.25	0.12
3	0.10	0.17	0.50	0.14
Average	0.13	0.19	0.42	0.15

Table 5-41. Flat intersection, back path error for straight navigation experiment.

Run	Average Error (in)	RMS Error (in)	Max Error (in)	Standard Deviation (in)
1	0.14	0.23	0.50	0.18
2	0.08	0.14	0.25	0.12
3	0.10	0.16	0.25	0.12
Average	0.11	0.18	0.33	0.14

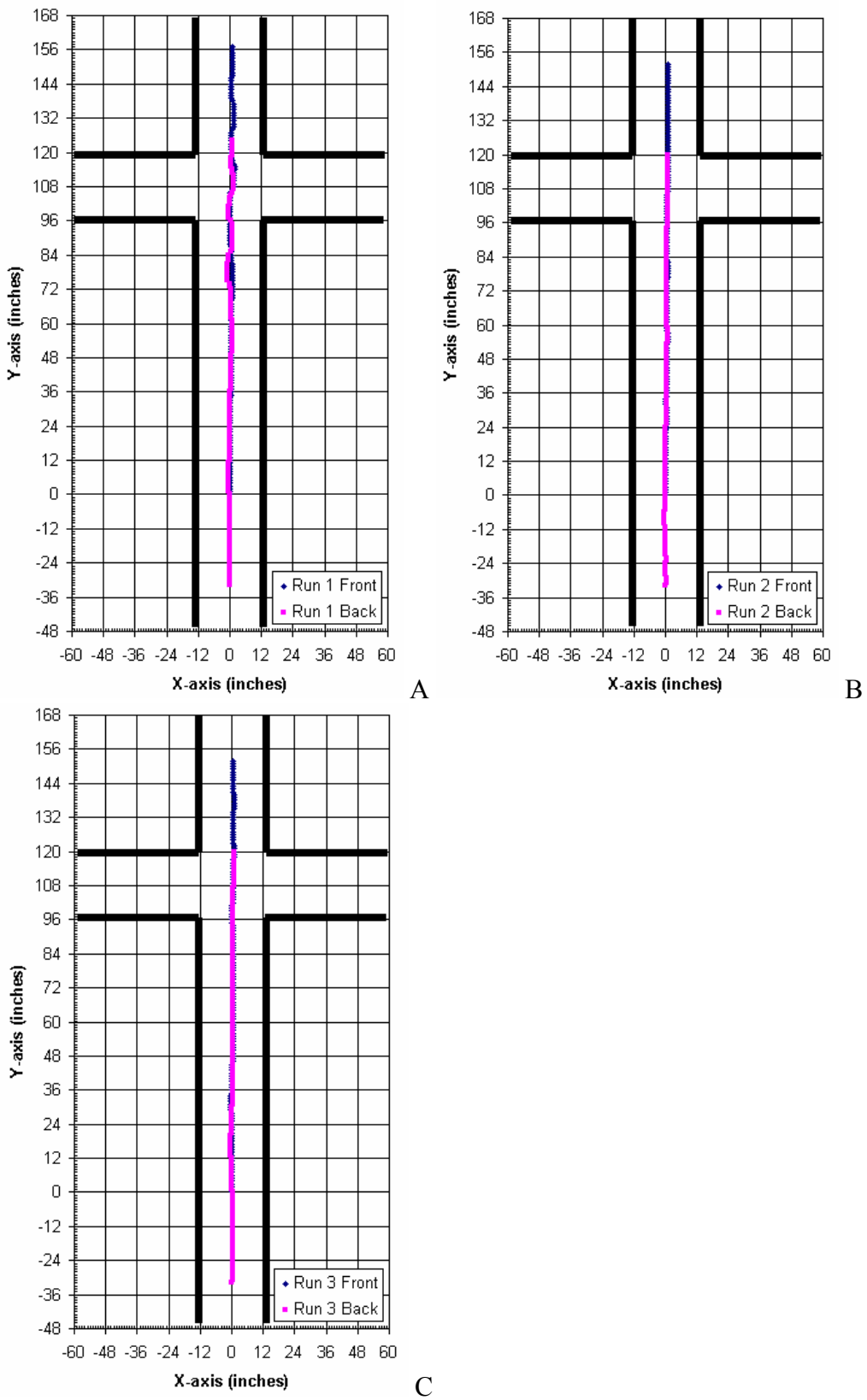


Figure 5-27. Recorded vehicle paths for plant intersection straight navigation experiment. A) Run 1. B) Run 2. C) Run 3.

Table 5-42. Plant intersection, front path error for straight navigation experiment.

Run	Average Error (in)	RMS Error (in)	Max Error (in)	Standard Deviation (in)
1	0.43	0.63	1.75	0.47
2	0.47	0.54	0.75	0.28
3	0.30	0.39	0.75	0.24
Average	0.40	0.52	1.08	0.33

Table 5-43. Plant intersection, back path error for straight navigation experiment.

Run	Average Error (in)	RMS Error (in)	Max Error (in)	Standard Deviation (in)
1	0.35	0.52	1.25	0.38
2	0.36	0.46	0.75	0.28
3	0.32	0.38	0.75	0.21
Average	0.34	0.45	0.92	0.29

CHAPTER 6
CONCLUSION

Based on the experimental results presented and discussed in Chapter 5, conclusions are drawn about the navigation algorithms, followed by a discussion of future work in the area.

Conclusion

Navigation algorithms were successfully developed for an autonomous greenhouse sprayer, enabling the vehicle to detect and navigate through intersections. Testing verified the accuracy of the vehicle's camera model, visual odometer, ability to detect intersections, and ability to navigate intersections. The camera model demonstrated the capability to relate image pixels to real-world coordinates with a maximum error of 3.70 inches at a distance 78.00 inches away, and minimum error of 0.08 inches at a distance of 36 inches away. The camera model error affected the accuracies of the visual odometer and intersection detection algorithms, but was accurate enough to allow the vehicle to successfully navigate through the intersection without major collision into the path edges or intersection corners.

The visual odometer showed accurate estimation of vehicle translation and rotation. Translation tests of the odometer in a lab environment gave an average error of 1.91 inches for a 12-inch forward translation and an average error of 4.88 inches for a 120-inch forward translation. Rotation tests of the odometer in a lab environment gave an average error of 1° for a 45° rotation about the vehicle z-axis and an average error of 8°

for a 180° of rotation about the vehicle z-axis. These were within an acceptable range, as demonstrated in the intersection navigation test for turning, allowing the visual odometer guide the vehicle to a position in the intersection suitable for the turn. Any positional errors in the location of the vehicle turning center were dealt with successfully by the path following algorithm after the turn, which corrected any offset error and guided the vehicle towards the center of the second path. Finally, tests completed on concrete, sand, and gravel demonstrated adaptability of the odometer on different ground surfaces that are common in greenhouses.

The intersection detection algorithm showed capability of detecting intersections in flat, structured path environments, and more natural, unstructured path environments. For a flat intersection lined with tape, beginnings of the intersection were detected with an average error of 0.66 inches when 36 inches from the intersection, 1.11 inches when 66 inches from the intersection, and under 2 inches when rotating the vehicle -8° to 8°, while positioned 48 inches from the intersection. For an intersection lined with plants, beginnings of the intersection were detected with an average error of 1.16 inches when 36 inches from the intersection, 6.04 inches when 66 inches from the intersection, and under 6 inches when rotating the vehicle -8° to 8°, while positioned 48 inches from the intersection. Ends of intersections were detected with a slightly higher average error. Many of these errors could be reduced through improvement of the camera model.

The overall navigation system proved capable of navigating through both the tape and plant-lined intersections. The vehicle followed instructions to make turns at the intersection, as well as drive straight through the intersection. For turns through the flat intersection the following results were obtained:

- The front of the vehicle averaged a path error of 0.45 inches along the first path center

- The vehicle used the visual odometer to guide itself to a desired position in the detected intersection to turn with an average turning center position error of 2.00 inches and average orientation error of 0.72° off the first path

- The vehicle completed the turn with an average orientation error of 2.86° off the second path center

- The front of the vehicle averaged a path error of 1.26 along the second path

For turns through the plant intersection the following results were obtained:

- The front of the vehicle averaged a path error of 0.46 inches along the first path center

- The vehicle used the visual odometer to guide itself to a desired position in the detected intersection to turn with an average turning center position error of 2.96 inches and average orientation error or 1.20° off the first path

- The vehicle completed the turn with an average orientation error of 4.44° off the second path center

- The front of the vehicle averaged a path error of 3.08 along the second path

Driving straight through the intersection yielded an average path error of 0.13 inches for the flat intersection and 0.40 inches for the plant intersection.

The thesis objectives stated in Chapter 1 developed for successful vehicle guidance in a greenhouse environment were met:

1. Path Following: The vehicle demonstrated the ability to follow the center of a 24-inch wide path. Smaller paths can also be followed with the same routine since both edges are still visible. The strategy for following a single edge was described in Chapter 3 and also implemented in the navigation system.

2. Intersection Detection: Both the beginning and end of intersections were detected for flat intersections lined with tape and intersections lined with potted plants two feet tall. The intersection detection algorithm calculates the beginning and end of the intersections independent of the height of objects at the corners of the intersection.

3. Intersection Navigation: The vehicle demonstrated the ability to turn at intersections, as well as drive through them.

Discussion and Future Work

This thesis addressed the use of machine vision for intersection detection and navigation of a robotic sprayer through a greenhouse. Technology critical to this goal were developed and discussed. Testing was carried out in setups made to simulate actual greenhouse environments. A next step in the testing process would be to run the vehicle in an actual greenhouse. Implementation of additional sensors into the current system should also be researched to provide a more robust navigation system, as well as enable the vehicle to navigate intersections if there is no path available for the visual odometer to track.

Further improvement can also be accomplished on the camera model and visual odometer. The camera model can be improved by reevaluating the extrinsic parameter geometry described in Chapter 3. Computational calculation of the extrinsic parameters can also be completed using techniques demonstrated by Bouguet (2004). The visual odometer can be improved by averaging vehicle translation and rotation over multiple points over several frames, as opposed to using just two features every frame. Utilization of techniques researched by Nistér et al. (2004) can allow the visual odometer to track features that aren't restricted to the ground plane.

The techniques investigated in this thesis can also be applied to robot navigation through orchards, citrus groves, and other corridor environments. The individual modules, such as the camera model and visual odometer, can also be used for various other autonomous navigation and machine vision applications.

LIST OF REFERENCES

Baba, M., Asada, N., Oda, A., & Migita, T. (2002). A Thin Lens Based Camera Model for Depth Estimation from Defocus and Translation by Zooming. Hiroshima, Japan: Department of Intelligent Systems, Hiroshima City University.

Barron, J., Fleet, D., & Beauchemin, S. (1994). Performance of Optical Flow Techniques. International Journal of Computer Vision, Vol. 12(1): 43-77.

Bartok, J. W. (1992, Feb.). Give Your Plants a Fast Start Use a Germination Unit. Greenhouse Newsletter.

Benson, E. R., Reid, J. F., & Zhang, Q. (2003). Machine Vision-Based Guidance System for an Agricultural Small-Grain Harvester. Transactions of the ASAE, ISSN 0001-2351, Vol. 46(4), 1255-1264.

Bouguet, J. (2004). Camera Calibration Toolbox for Matlab [Online Website]. Aug. 2004. Available WWW: http://www.vision.caltech.edu/bouguetj/calib_doc

Carelli, R., Soria, C., Nasisi, O., & Freire, E. (2002, Nov.). Stable AGV Corridor Navigation with Fused Vision-Based Control Signals. IECON 02, Vol. 3, 2433-2438.

Coulter, R. C. (1992, Jan.). Implementation of the Pure Pursuit Path Tracking Algorithm (Technical Report CMU-RI-TR-92-01). Pittsburgh, PA: Robotics Institute, Carnegie Mellon University.

Crane, C. & Duffy, J. (1998). Kinematic Analysis of Robot Manipulators. New York, New York: Cambridge University Press, 9.

Csetverikov, D. (2004). Basic Algorithms for Digital Image Analysis: a course. Budapest, Hungary: Institute of Informatics Eötvös Loránd University. Aug. 2004. Available WWW: http://visual.ipan.sztaki.hu

Dai, B., Liu, J.P., & Chang, W.S. (1995, June). Real-time Obstacle Detection via Subsequent 2D Images. Proceedings of SPIE, Vol. 2463, 255-263.

DeSouza, G. N., & Kak, A. C. (2002, Feb.). Vision for Mobile Robot Navigation: A Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 2, 237-267.

Fehr, B. W., & Gerrish, J. B. (1995). Vision-Guided Row-Crop Follower. Appl. Eng. Agric., Vol. 11 (4), 613-620.

Fernandez, J., & Casals, A. (1997, Sept. 7-11). Autonomous Navigation in Ill-Structured Outdoor Environments. Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol. 1, 395-400.

Figliola, R. S., & Beasley, D. E. (2000). Theory and Design for Mechanical Measurements. New York, New York: John Wiley & Sons, Inc.,122.

Fischler, M. A., & Bolles, R. C. (1981, June). Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Communications of the ACM 24, 6.

Gray, S. A. (2001). Planning and Replanning Events for Autonomous Orchard Tractors. Master Thesis. Logan, Utah: Utah State University.

Hague, T., Marchant, J. A., & Tillet, N. D. (1997, Apr. 20-25). Autonomous Robot Navigation for Precision Horticulture. Proceedings of the 1997 IEEE International Conference on Robots and Automation, Vol. 3, 1880-1885.

Hardy, T. (2003, July). Automated Harvesting Research Merits Grower Support. The Fruit Growers News. Great American Publishing.

Have, H., S. Blackmore, Keller, B., Fountas, H., Nielsen, S., & Theilby, F. (2002). Autonomous Weeders for Christmas Tree Plantations- A Feasibility Study. Pesticide Research 59. Denmark: Danish Environmental Protection Agency, Danish Ministry of the Environment.

Hellström, T. (2002). Autonomous Navigation for Forest Machines: A Pre-Study. Umeå, Sweden: Department of Computing Science, Umeå University.

Hellström T., & Ringdahl, O. (2002). Follow the Past – A Path Tracking Algorithm for Autonomous Forest Vehicles. Umeå, Sweden: Department of Computing Science, Umeå University.

Jochem, T., Pomerleau, D., & Thorpe, C. (1996). Vision Based Intersection Navigation. Pittsburgh, PA: The Robotics Institute, Carnegie Mellon University.

Kessler, J. R. (1998, May). Hobby Greenhouse Construction. Alabama Cooperative Extension System, ANR-1105, Alabama A & M and Auburn Universities.

Kundur, S.R., & Raviv, D. (1999, Feb.). Novel Active Vision-Based Visual Threat Cue for Autonomous Navigation Tasks. Computer Vision and Image Understanding, Vol. 73, No. 2, 169-182.

Lavest, J.M., Rives, G., & Dhome, M. (1993, Apr.) Three-Dimensional Reconstruction by Zooming. IEEE Tranasactions on Robotics and Automation, Vol. 9, No. 2, 196-208.

Lavest, J.M., Delherm, C., Peuchot, B., & Daucher, N. (1997, June). Implicit Reconstruction by Zooming. Computer Vision and Image Understanding, Vol. 66, No. 3, 301-315.

Leavy, P. (2002, Oct. 4). Citrus Shakedown. Tampa Bay, FL: The Business Journal.

Matsumoto, Y., Ikeda, K., Inaba, M., & Inoue, H. (2000, Nov.). Exploration and Navigation in Corridor Environment Based on Omni-View Sequence. Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol. 2, 1505-1510.

Morimoto, E., Suguri M., & Umeda, M. (2002, Dec.). Obstacle Avoidance System for Autonomous Transportation Vehicle based on Image Processing. Agricultural Engineering International: the CIGR Journal of Scientific Research and Development, Manuscript PM 01 009, Vol. 4.

Nistér, D., Naroditsky, O., & Bergen, J. (2004). Visual Odometry. Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, 652-659.

Ollis, M., & Stentz, A. (1997, Sept. 7-11). Vision-Based Perception for an Automated Harvester. Proceedings of the 1997 International Conference on Intelligent Robots and Systems, Vol. 3, 1838-1844.

Ollis, M., & Stentz, A. (1996). First Results in Vision-Based Crop Line Following. Proceedings of the 1996 IEEE International Conference on Robotics and Automation, Vol. 1, 951-956.

Pilarski, T., Happold, M., Pangels, H., Ollis, M., Fitzpatrick, K., & Stentz, A. (1999). The Demeter System for Automated Harvesting. Pittsburgh, PA: The Robotics Institute, Carnegie Mellon University.

Pollefeys, M. (2004, Spring). Visual 3D Modeling from Images. Tutorial Notes. Chapel Hill, NC: University of North Carolina.

Pomerleau, D. (1995). Neural Network Vision for Robot Driving. The Handbook of Brain Theory and Neural Networks, M. Arbib. Pittsburgh, PA: The Robotics Institute, Carnegie Mellon University.

Rasmussen, C. (2002, May 11-15). Combining Laser Range, Color, and Texture Cues for Autonomous Road Following. Proceedings of the ICRA 2002 IEEE International Conference on Robotics and Automation, Vol. 4, 4320-4325.

Rasmussen, C. (2003). Road Shape Classification for Detecting and Negotiating Intersections. Newark, DE: Department of Computer & Information Sciences, University of Delaware.

Reid, J. F. (1998, Sept.). Precision Guidance of Agricultural Vehicles. Presented at JSME Meeting, Sapporo, Japan. UILU-ENG-7031.

Reid, J. F., Zhang, Q., Noguchi, N., & Dickson, M. (2000). Agricultural Automatic Guidance Research in North America. Computers and Electronics in Agriculture, Vol. 25, 155-167.

Schnelle, M. A., & Dole, J. M. (1991, Jan.). Greenhouse Floors and Benches. OSU Extension Facts, F-6703. OK: Oklahoma Cooperative Extension Service, Oklahoma State University.

Singh, S. (2004). Autonomous Robotic Vehicle for Greenhouse Spraying. Master's Thesis. Gainesville, FL: University of Florida.

Singh, S., & Subramanian, V. (2004, Oct.). Autonomous Greenhouse Sprayer Vehicle using Machine Vision and Ladar for Steering Control. Proceedings of the 2004 ASAE Automation Technology for Off-road Equipment, St. Joseph, MI, 79-90.

Stevens, A. B., Stevens, S., Albrecht, M. L., & Gast, K. L. (1994, Jun.). Starting a Greenhouse Business: A Commercial Growers Guide. Manhattan, KS: Cooperative Extension Service, Kansas State University.

Tucker, D. P. H., Wheaton, T. A., & Muraro, R. P. (1994, Jun.). Citrus Tree Pruning Principles and Practices. Florida Cooperative Extension Services, University of Florida (Fact Sheet HS-144).

University of Florida (2004). University of Florida Unmanned Vehicles: Vehicle Control [Online Web site]. Jun. 2004. Available WWW: http://www.me.ufl.edu/~webber/web1/pages/research_areas/vehcile_control.htm

Wit, J. S. (2000). Vector Pursuit Path Tracking for Autonomous Ground Vehicles. Ph.D. Dissertation. Gainesville, FL: University of Florida.

BIOGRAPHICAL SKETCH

Paulo Younse received his bachelor's degree in mechanical engineering with a mechatronics focus from California Polytechnic State University in San Luis Obispo, California. In August 2003, he joined the Master of Engineering program at the University of Florida in the Agricultural and Biological Engineering Department with a research focus on autonomous vehicle navigation for agricultural robots.